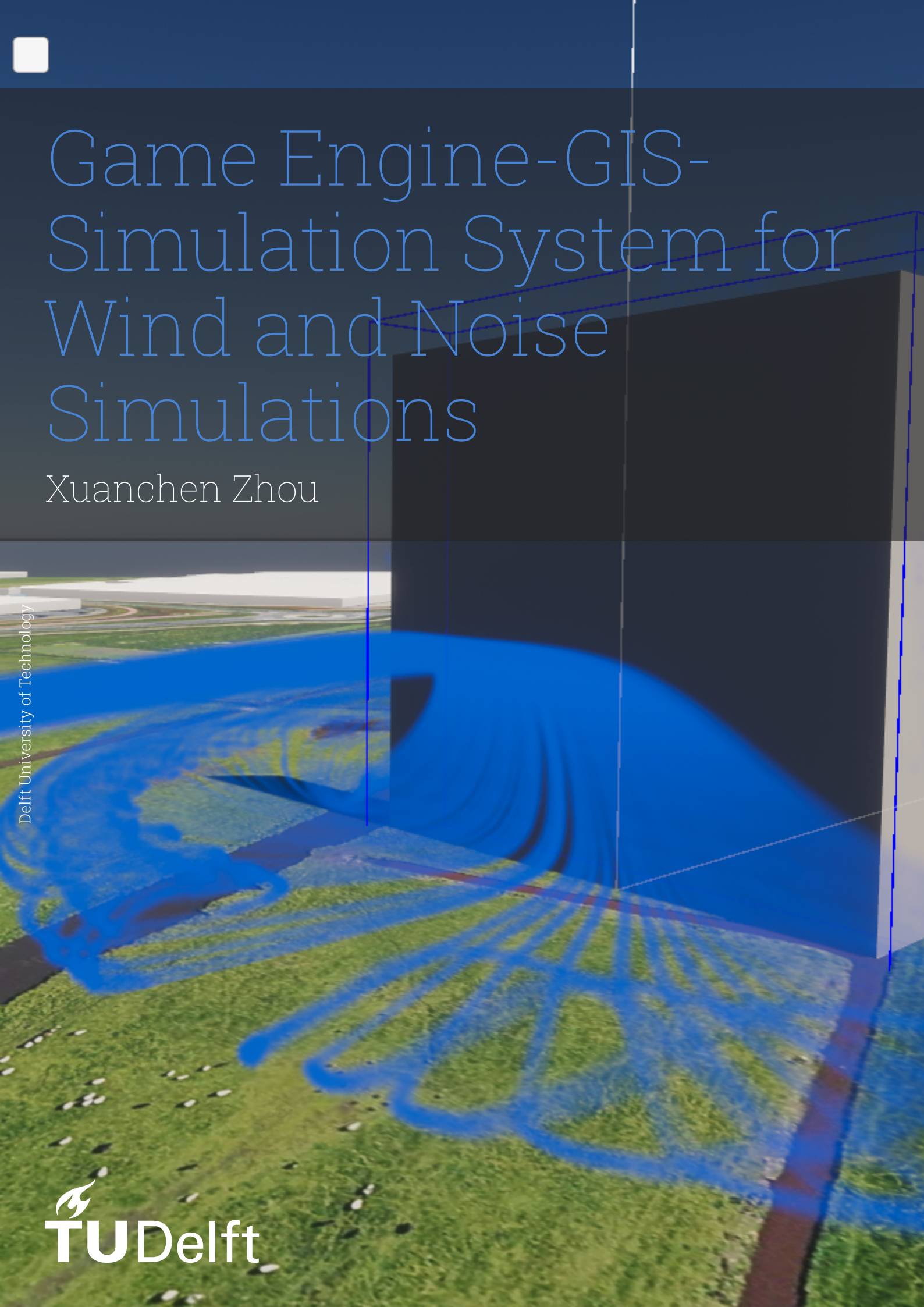




Game Engine-GIS- Simulation System for Wind and Noise Simulations

Xuanchen Zhou

Delft University of Technology

Game Engine-GIS-Simulation System for Wind and Noise Simulations

by

Xuanchen Zhou

Promoter:	Prof.dr.ir. Peter van Oosterom
Daily Supervisor:	Dr. Azarakhsh Rafiee
Faculty:	Faculty of Architecture and the Built Environment

Summary

Wind and noise are important environmental factors that significantly affect the quality of human life in urban areas. It is therefore important to consider them in the urban planning process. This PhD project aims to leverage the capabilities of modern game engine and geographical information system (GIS) to develop a high-performance, interactive framework for urban wind and noise simulation.

The project is organized into several research stages. The first stage focuses on building a game engine-GIS-computational fluid dynamics (CFD) triplet for urban wind simulation. The objective is to improve computational efficiency by accelerating the CFD computation with compute shaders, while incorporating GIS, Geo-DBMS and game engine functionalities to enhance interactivity and visualization of the wind simulator.

Building upon this foundation, the second stage investigates the application of adaptive mesh refinement (AMR) to further improve the efficiency of urban wind simulation. The author will focus on addressing the current lack of best practice guidelines for AMR in urban wind modelling, identifying AMR strategies suitable for compute shaders, and evaluating their accuracy and efficiency relative to conventional static mesh refinement approaches.

The third stage focuses on improving urban noise simulation by building a game engine-GIS-computational acoustics triplet. First, compute shaders will be employed to accelerate refraction-aware noise ray tracing. Then, wind and noise simulators will be coupled to investigate the benefits of incorporating simulated wind data in noise ray refraction simulation. The potential of ray tracing shaders will also be explored to further accelerate noise simulation, especially when atmospheric refraction can be neglected.

The fourth stage focuses on improving the convergence rate of stochastic noise ray tracing. Specifically, the research will investigate the application of multiple importance sampling (MIS) to noise ray tracing framework, and explore the potential integration of MIS with current refraction and diffraction models.

The final stage focuses on a comprehensive evaluation of the developed and improved urban wind and noise simulator, against the state-of-the-art open source wind and noise simulators. Quantitative comparisons will be conducted with respect to computational efficiency and accuracy, while qualitative assessments will examine usability and interactivity through analytical evaluation and user studies.

Contents

Summary	i
Nomenclature	iv
1 Introduction	1
1.1 Impact of urban wind and noise	1
1.2 Research status and research gap in urban wind analysis	1
1.3 Research objectives for urban wind simulation	2
1.4 Research status and research gap in urban noise simulation	3
1.5 Research objectives for urban noise simulation	3
1.6 Research questions	4
1.7 Hypotheses	5
2 Background	7
2.1 Urban wind analysis	7
2.1.1 Experimental approaches	7
2.1.2 Simulation approaches	8
2.1.3 CFD in urban wind simulation	9
2.1.4 CFD adaptive mesh refinement	10
2.2 Urban noise simulation	11
2.2.1 Wave-based methods	12
2.2.2 Geometrical acoustics	12
2.2.3 Energy-based methods	13
2.2.4 Noise propagation in urban region	13
2.2.5 Sound ray refraction models	14
2.2.6 Ray trace sampling strategies	15
2.3 Game engine	17
2.3.1 Parallel computation	17
2.3.2 Ray-mesh intersection	18
2.3.3 User interaction	18
2.3.4 Visualization	18
2.4 Geographical information system	19
2.4.1 Use GIS for urban wind and noise simulation	19
2.4.2 Leveraging Geospatial Database Management Systems for visualization	19
2.4.3 Limitations of traditional GIS frameworks	19
2.4.4 Typical GIS-simulation interoperability pipeline	20
3 Methodology	21
3.1 Improving usability and efficiency of wind simulator through game engine and GIS integration	22
3.1.1 Coupling CFD with a game engine	22
3.1.2 Enhancing usability by coupling with GIS functionalities	24
3.2 Improving efficiency of wind simulator through adaptive mesh refinement	24
3.2.1 Implementing AMR algorithm in compute shader	25
3.2.2 Develop grid refinement criteria	25
3.2.3 Validation and testing	25
3.3 Improving accuracy and efficiency of noise simulator using wind simulator and game engine	26

3.3.1	Accelerating ray tracing calculations with game engine	26
3.3.2	Coupling with the wind solver	27
3.4	Improving accuracy and efficiency of noise simulator using multiple importance sampling	27
3.4.1	Coupling MIS with refraction models	28
3.4.2	Coupling MIS with diffraction models	28
3.5	Performance and user-centered evaluation	28
3.5.1	Quantitative evaluation	28
3.5.2	Qualitative evaluation	29
3.5.3	User-centric experiment design	29
4	Preliminary results	30
4.1	Development of a compute shader-based large eddy simulation solver	30
4.1.1	Solver implementation	30
4.1.2	Solver validation	30
4.1.3	Solver performance evaluation	31
4.2	Coupling the urban wind field simulator with real world scene	31
4.2.1	Scene voxelization	31
4.2.2	Cesium integration	32
4.2.3	KNMI meteorological data integration	33
4.3	Visualization	33
4.4	Asynchronous data exportation	34
4.5	Web visualizer	34
5	Plan	37
5.1	Supervision and project team	37
5.2	Resources and tools	37
5.3	Publication plan	38
5.4	Conference plan	38
5.5	Master's thesis supervision plan	39
5.6	Graduate school course planning	39
5.7	Project timeline	39
5.8	Self-reflection of the first year	40
	References	43
A	Appendix A	50
B	Appendix B	59

Nomenclature

Abbreviations

Abbreviation	Definition
A-BRDF	Acoustic Bidirectional Reflectance Distribution Function
AIJ	Architectural Institute of Japan
AMR	Adaptive Mesh Refinement
BVH	Bounding Volume Hierarchy
CFD	Computational Fluid Dynamics
END	Environmental Noise Directive
FDTD	Finite-Difference Time-Domain
FIFO	First-In-First-Out
GA	Geometrical Acoustics
Geo-DBMS	Geographical Database Management System
GIS	Geographical Information System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
GTD	Geometrical Theory of Diffraction
KNMI	Royal Netherlands Meteorological Institute
LEE	Linearized Euler Equation
LES	Large Eddy Simulation
LVNL	Luchtverkeersleiding Nederland
MIS	Multiple Importance Sampling
NDW	Nationaal Dataportaal Wegverkeer
NEE	Next Event Estimation
NS	Navier-Stokes
NWB	Nationaal Wegen Bestand
PDF	Probability Density Function
PSTD	Pseudospectral Time-Domain
ROI	Region of Interest
RT	Ray Tracing
SIMPLE	Semi-Implicit Method for Pressure-Linked Equations
SubRQ	Sub Research Question
VFX	Visual Effect
VR	Virtual Reality

Introduction

1.1. Impact of urban wind and noise

Wind and noise are both important environmental factors that significantly affect human life in urban areas. As urbanization continues to accelerate worldwide, cities are becoming increasingly dense and complex, which leads to growing environmental challenges related to urban microclimates, airflow behavior, pollutant dispersion, and environmental noise. Consequently, understanding and predicting urban wind and noise environments is of highly importance in urban planning process.

Urban wind environment significantly impacts various aspects of human society and city life. If utilized properly, urban wind can serve as potential renewable energy resources [44], mitigate the urban heat island effect [41, 99] and enhance the dispersion of atmospheric pollutants [60, 89]. Natural ventilation driven by urban airflow can also reduce building energy consumption [40]. However, adverse wind conditions may also introduce significant risks. Strong wind around high-rise buildings can threaten structural safety [29], damage trees and urban infrastructure [12], and negatively affect pedestrian comfort and safety [12, 32]. Consequently, understanding urban wind behavior is essential for urban planning, public safety management and environmental assessment.

In parallel with the challenges of urban wind environments, noise pollution has also become an increasingly severe environmental issue due to rapid urbanization and the expansion of transportation[39]. Road traffic, railways, aviation, industrial and community activities are major contributors to environmental noise in modern cities [100]. Numerous studies have shown that long-term exposure to environmental noise can negatively affect human health, leading to sleep disturbance, cardiovascular disease, psychological stress, and reduced quality of life [13, 108]. Environmental noise also impacts people's normal lives, work efficiency, and residential satisfaction [111, 48]. Consequently, various international and regional regulations and guidelines have been established to assess and mitigate urban noise pollution [9, 13, 108]. It is thus important to develop physically accurate noise propagation models, in order to support the noise regulations formulation process and urban planning, as well as provide data to improve urban living environments.

Due to the significant societal, environmental, and health impacts of urban wind and noise, it is crucial to consider these factors during the early stages of urban planning and design [45, 46].

1.2. Research status and research gap in urban wind analysis

A wide range of methods have been developed for urban wind analysis. Experimental approaches mainly include field measurements and wind tunnel experiments, whereas computational approaches include semi-empirical methods and computational fluid dynamics (CFD)-based methods. Together with the rapid development of modern computing hardware, CFD-based methods have attracted increasing

attention due to its favorable balance between computational accuracy and physical fidelity.

Academic urban wind simulation research primarily utilizes professional CFD software [102]. Although being highly accurate and versatile, these software packages are designed for general-purpose CFD applications, which result in complex scene setup and solver setup stages before simulation. Such configurations demand professional knowledge to conduct, therefore raising the barrier for involving the wind modeling into urban planning process [36].

Several plugins of geometric modeling software have been developed specifically for urban wind environment simulation [47]. By utilizing the geometric modeling abilities of geometric modeling software and the provision of pre-defined simulation settings, the operational complexity of simulation preparation stage can be dramatically reduced. Nevertheless, these plugins are predominantly CPU-based and only support steady-state solvers. This characteristic poses challenges in simulating transient phenomena within the urban wind environment, such as vortex shedding on the leeward side of a building, wind gusts and pollutant dispersion [103, 97]. A usability & efficiency comparison of existing urban wind simulation CFD pipelines is demonstrated on Fig. 1.1 (left).

In order to enhance computational efficiency, adaptive mesh refinement (AMR) has been proposed in CFD. This technique enables localized refinement of subregions within the computational domain based on predefined criteria, thereby reducing computational cost in less critical areas. It is widely recognized that AMR can significantly improve performance while maintaining comparable accuracy [110]. However, comprehensive best-practice guidelines for AMR in the context of urban wind simulation are still lacking [103, 65, 27]. Consequently, its adoption in urban wind simulation remains limited [37, 84].

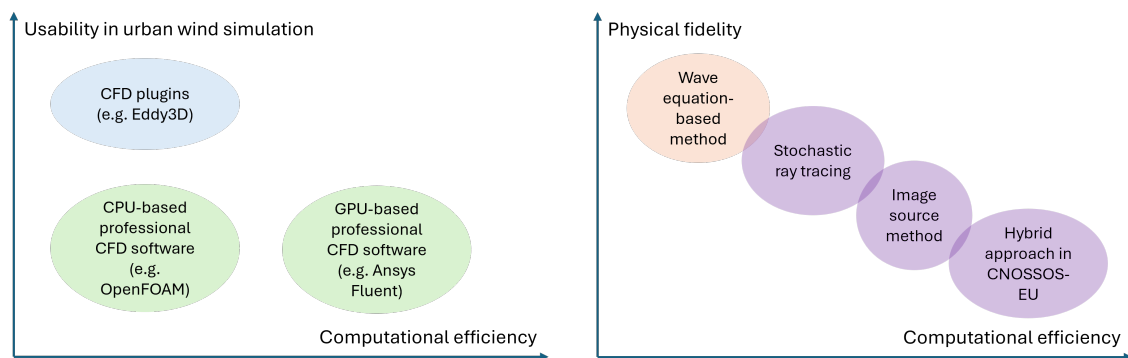


Figure 1.1: (Left) Comparison between usability & computational efficiency of different existing wind solvers. (Right) Comparison between physical fidelity & computational efficiency of different noise simulation models. Same color means same type of solvers (left) or models (right). The energy-based noise simulation methods are not the focus of this project, it is thus neglected on this figure.

1.3. Research objectives for urban wind simulation

This project aims to address the aforementioned challenges. In the first phase, the author aims to design and develop an urban wind simulator that achieves the following objectives:

1. High usability: eliminating the need for complex pre-processing procedures.
2. High performance: accelerate the solver to fulfill the requirements of the dynamic decision-making process in urban planning.

To achieve these objectives, existing methods must be improved in terms of both usability and computational efficiency. Usability will be enhanced by leveraging the functionalities of game engine and integrating GIS. Performance improvements will be realized through the use of GPU programming supported by game engines. The goal of this phase is to develop a user-friendly and high-performance urban wind simulator to support urban wind-related decision making.

In the second phase, the author will investigate the applicability of AMR in urban wind simulation. Given the lack of best-practice guidelines and related studies, the following questions will be addressed:

1. Can a best practice guideline that can help with selecting effective refinement criteria in urban wind simulation be concluded?
2. Which algorithms are most suitable for compute shader-based AMR? and to what extent can AMR improve the accuracy and efficiency of urban wind simulation?

The objective of this phase is to further enhance solver efficiency through the application of AMR.

1.4. Research status and research gap in urban noise simulation

In parallel with urban wind analysis, various approaches have also been developed for urban noise simulation. These methods can generally be categorized into wave-based methods, geometric-based methods and energy based methods, with geometric-based methods further divided into several sub-categories [34, 58, 79, 10, 59, 48] (Fig. 1.1). Wave based methods provide high physical accuracy, but are computationally prohibitive for large-scale urban environments. Energy based methods rely on diffusion assumptions for sound propagation, which is not accurate in many urban cases. Comparatively, geometric-based methods have been widely adopted in urban noise modeling due to its favorable balance between computational efficiency and accuracy. Among these geometric-based methods, stochastic ray tracing is considered one of the most physically accurate noise simulation models for large scene simulation. However, its relatively high physical fidelity is accompanied by substantial computational cost, which limits its applicability in urban noise simulation.

One critical factor affecting both accuracy and efficiency of ray-tracing based noise simulation is the treatment of sound ray refraction. In the presence of wind and temperature gradients, sound rays bend as they propagate through the atmosphere. To account for this effect, various modeling approaches have been proposed. The most accurate approaches discretize ray path into multiple segments, with the propagation direction in each segment determined by local wind and temperature conditions [58, 59]. These methods can achieve high physical fidelity. However, they require solving ordinary differential equations or performing geometric approximations at each step, leading to significantly higher computational costs [88, 42, 58]. Although some studies have leveraged GPUs for urban noise ray tracing, existing implementations are largely limited to straight-line propagation models [117]. The more computationally intensive segmented approaches have not yet been effectively accelerated. Moreover, although recent studies have shown that dedicated ray tracing hardware in GPU can provide substantial performance gains compared to conventional GPU-based approaches, current methods in urban noise simulation still rely on general-purpose GPU cores [57].

Moreover, while multiple numerical methods have been proposed to model the influence of wind fields on noise propagation, few works have validated them under realistic urban wind conditions. This research also aim to discuss whether these methods are applicable within simulated urban wind fields, and what algorithmic corrections might be needed.

Another factor that significantly affects the computational efficiency of stochastic ray tracer is the reflection ray sampling strategy. Existing stochastic ray-tracing approaches mainly rely on traditional importance sampling and diffuse ray algorithms. However, both strategies exhibit limitations in certain scenarios. In computer graphics, multiple importance sampling has been developed to address similar sampling inefficiencies and high variance issues. Despite its success in graphics rendering, this technique has not yet been explored in the context of urban sound propagation. As a result, the effect of its potential integration with refraction and diffraction modeling remains largely unknown.

1.5. Research objectives for urban noise simulation

Due to the aforementioned limitations, this project focuses on two main aspects of urban noise simulation. First, the author aims to develop a GPU-accelerated urban noise simulator to address the current lack of efficient implementations of GPU-accelerated refraction-aware urban ray tracing noise simulation, while also improving usability and enabling coupling with an urban wind simulator. The specific objectives are as follows:

1. Improve computational efficiency by leveraging both compute shaders and ray tracing shaders in a game engine.
2. Couple the noise simulator with the urban wind simulator to validate the refraction models under realistic flow conditions and refine these models if necessary.

Second, the author focuses on algorithmic aspects of urban noise simulation, aiming to improve the underlying computational models. In particular, the author will explore how multiple importance sampling can be incorporated into ray tracing-based noise simulation, and to combine it with existing refraction and diffraction models.

1.6. Research questions

Based on the research objectives outlined above, the central research question of this project can be defined as:

To what extent can the integration of game engines and GIS, combined with targeted algorithmic optimizations, enhance the overall performance of urban wind and noise simulators?

The main research question can be divided into five sub research questions (SubRQ):

1. How to utilize a game engine-GIS-CFD triplet to improve the computational efficiency and usability of urban wind simulation?

GIS functionalities provide connection to high-fidelity data for urban geometry, including buildings and terrain, as well as meteorological data required for wind simulation. Game engines offer highly parallel processing capabilities through compute shaders, together with integrated user interaction, scene modification and advanced visualization tools. This subquestion investigates how can these capabilities be effectively combined to develop highly usable and computationally efficient urban wind simulators.

2. To what extent can the computational efficiency of urban wind simulator be improved through adaptive mesh refinement in game engine?

Currently, the application of adaptive mesh refinement (AMR) in urban wind simulation lacks established best-practice guidelines and is rarely used in practical analyses. This sub-question aims to expand the utilization of AMR in urban wind simulation, establish practical guidelines for its use, and systematically evaluate its impact on both accuracy and computational efficiency.

3. How can an integrated framework combining game engine, GIS, CFD and CA optimize the physical fidelity, computational efficiency and usability of ray tracing-based urban noise simulator?

This subRQ investigates how GIS and game engine can be integrated into acoustic simulation framework to improve computational efficiency, usability, and visualization capabilities. It will focus on accelerating ray tracing-based urban noise simulation system utilizing compute shaders, and also explore the potential usage of ray tracing shaders. It will also focuses on coupling the noise simulator with the developed urban wind simulator, evaluating the performance of existing refraction models under simulated urban wind fields, and refining these models when necessary to improve physical fidelity and computational robustness.

4. To what extent can the convergence rate and accuracy of ray tracing-based urban noise simulator be improved through multiple importance sampling?

Identifying valid propagation paths from a sound source to a receiver is computationally challenging, particularly in large and complex urban environments. Traditional ray tracing approaches are predominantly unidirectional, where each emitted ray has a relatively low probability of reaching the receiver. This sub-question explores the use of multiple importance sampling to guide ray propagation by combining physically-based reflection sampling with receiver-directed sampling strategies, thereby improving both efficiency and accuracy.

5. How does the developed system perform in terms of efficiency, and how do users evaluate it in terms of interactivity and satisfaction comparing to state-of-the-art urban wind and noise simulators?

After the whole urban wind and noise simulation system is built, the last SubRQ focuses on the evaluation of our system. We will focus on the quantitative evaluation of the computational efficiency, comparing with state-of-the-art open-source solvers, and user reflections about their satisfaction and feedback on the system.

To systematically evaluate the proposed research questions and the developed simulation system, the following Key Performance Indicators (KPIs) are defined:

- **Computational efficiency:** given a specific modern hardware configuration, the computational time required for an algorithm or solver to achieve a target level of simulation accuracy.
- **Usability:** the degree of human effort and manual intervention required to setup and run an urban simulation scenario.
- **Physical fidelity:** the physical accuracy of the underlying computational models. A method achieves higher physical fidelity by minimizing assumptions and explicitly resolving the governing equations (e.g., employing segmented ray tracing for atmospheric refraction instead of straight-line approximations).
- **Performance:** comprehensive metric evaluating all the above indicators.

1.7. Hypotheses

The proposal of the research subquestions depends on several hypothesis, which will be validated during SubRQs 1-4, and summarized in SubRQ 5:

- **Hypothesis 1:** Game engine's high performance modules, including compute and ray tracing shaders, can be used to accelerate the computational efficiency of CFD and ray tracing-based CA solvers.

This hypothesis is mainly investigated in SubRQ 1 and 3. It will be evaluated through comparing the computational efficiency of compute and ray tracing shaders accelerated wind and noise solvers, against the non-accelerated solvers.

- **Hypothesis 2:** Game engine's user interaction modules and visualization modules can be used to enhance the interoperability and usability of CFD and ray tracing-based CA solvers.

This hypothesis is mainly investigated in SubRQ 1 and 3. It will be evaluated in the development process of urban wind and noise solvers. The interactivity of the developed solver will be compared with state-of-the-art urban wind and noise solvers.

- **Hypothesis 3:** GIS functionalities, such as different data sources, can enhance the usability of urban wind and noise simulator.

This hypothesis is mainly investigated in SubRQ 1 and 3. It will be evaluated in the development process of urban wind and noise solvers. The usability of the developed solver will be compared with state-of-the-art urban wind and noise solvers.

- **Hypothesis 4:** Adaptive mesh refinement can significantly increase the computational efficiency in game engine-based urban wind simulator.

This hypothesis is mainly investigated in SubRQ 2. It will be evaluated through the computational efficiency comparison between adaptive mesh refinement and traditional static mesh refinement in urban wind simulation.

- **Hypothesis 5:** The physical fidelity of an urban noise simulator can be significantly increased by coupling with a wind simulator.

This hypothesis is mainly investigated in SubRQ 3. It will be evaluated through the accuracy comparison between urban noise simulator with / without wind simulator enhancement.

- **Hypothesis 6:** The computational efficiency of an urban noise simulator can be significantly increased using compute shader and ray tracing shader.

This hypothesis is mainly investigated in SubRQ 3. It will be evaluated through the computational efficiency comparison between urban noise simulator with / without compute shader and ray tracing shader acceleration.

- **Hypothesis 7:** Multiple importance sampling can significantly increase the convergence rate and computational efficiency of ray tracing-based acoustic simulator.

This hypothesis is mainly investigated in SubRQ 4. It will be evaluated through the convergence rate and computational efficiency between multiple importance sampling strategy and other sampling strategies.

2

Background

2.1. Urban wind analysis

In densely built urban areas, complex building geometries can generate highly complicated airflow phenomena, including turbulent flow structures, wind tunnels, wake regions and localized strong-wind areas. These complex interactions between airflow and urban geometry make it essential to accurately understand urban wind behavior through measurement or simulation. Consequently, urban wind analysis plays an important role in sustainable urban design, environmental assessment, and public safety management. Generally, the assessment of urban wind effects can rely on multiple different techniques, which can be roughly divided into four categories: field measurement, wind tunnel experiments, diagnostic methods and computational fluid dynamics.

2.1.1. Experimental approaches

The most straightforward approach of analyzing urban wind is to collect data in real world. Field measurement method conducts experiments in real atmospheric conditions, and can therefore consider all the physical phenomena, such as compressibility, heat effect and humidity without approximation (Fig. 2.1). However, the measurement quality relies on the sampling equipment, which is expensive and time-consuming to set up. And real world experiment means that only limited sample points can be measured instead of the whole field. Most importantly, real world measurement means that it is impossible to change the inlet profile and test the wind speed in assumed extreme conditions. And continuously changing external boundary conditions make it impossible to repeat experiments under identical flow conditions [50].



Figure 2.1: Equipments to collect real world wind data, (left) a drone with 2D sonic anemometers, (right) a 3D sonic anemometer on the top of a building. (sources: [56])

To make the measurements controllable and repeatable, wind tunnels are used together with reduced-

scale building models to produce the wind behavior (Fig. 2.2). The boundary conditions can be chosen and repeated in wind tunnel experiments, which makes it possible to test the wind behavior under assumed wind profile and imaginary designs. But this method also suffers from the problem of limited number of measure points and significant setup cost. Its another problem is the loss of physical similarity. Due to the reduced physical scale, the Reynolds number of wind tunnel experiments is often far smaller than real wind conditions. The differences in Reynolds number can lead to different physical phenomena, making the modelling not accurate enough.

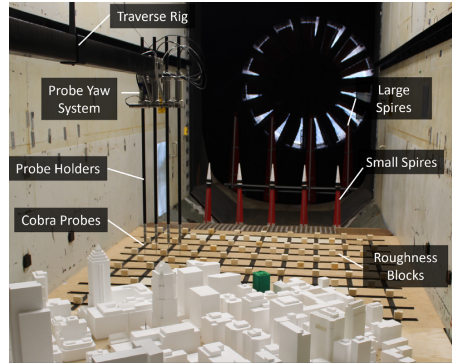


Figure 2.2: Components in a typical building wind tunnel. (source: [56])

2.1.2. Simulation approaches

Semi-empirical approaches

In order to reduce the cost of wind analysis, simulation-based methods are applied. The diagnostic mass-consistent flow field model [80, 98] has been used in predicting wind field using small amount of computational resources. Its general idea is to perform a constraint optimization problem with the goal of making the velocity field divergence free while not deviating too much from the given original field. This method can produce tolerable velocity field if a good original field is given, but can lead to totally physically incorrect result if the original field is far from the actual field. As a consequence, it cannot capture some wind features that are important in urban wind simulation, such as downwind separation and channel effect [30, 80].

Computational fluid dynamics approaches

Together with the development of modern computers, advanced modelling techniques have been more and more widely used in urban wind simulation. By applying the continuity equations and choosing an appropriate constitutive model, Navier Stokes (NS) equations that can accurately describe the motion of fluid can be derived:

$$\begin{aligned} \nabla \cdot \mathbf{u} &= 0, \\ \frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) &= -\frac{1}{\rho} \nabla p + \nabla \cdot \left(\nu \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right) \right). \end{aligned} \quad (2.1)$$

Eq. 2.1 demonstrates the incompressible form of NS equation. The incompressible assumption is often made when the characteristic velocity simulated $u \ll c_s$, where c_s is the sound speed. ρ , $\mathbf{u}$, and p denote the density, velocity and pressure, respectively. ν denotes the kinematic viscosity.

By discretizing Eq. 2.1 and choosing appropriate numerical schemes, NS equations can be solved on computers. The resulting method is called computational fluid dynamics (CFD) methods (Fig. 2.3). As its name and essential suggest, it can be highly accurate and able to solve the complex flow phenomena between the buildings [17]. CFD methods can solve problems introduced by the previous methods, it is much cheaper comparing to experimental based methods, as no physical wind tunnel and measurement equipments need to be used. Also the whole simulation field can be obtained, instead of limited number of sample points. By performing experiments on the computer, no reduced scale models needs to be used, and the physical identity behavior can be obtained.

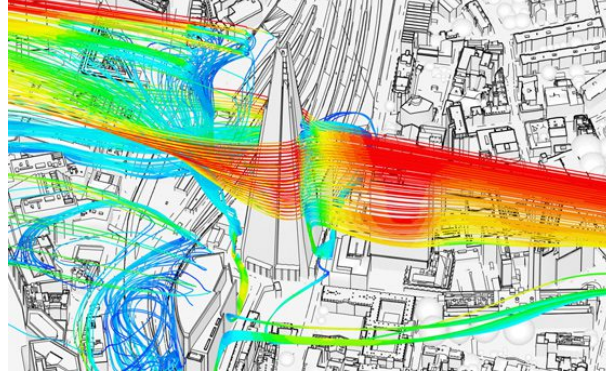


Figure 2.3: An example of CFD urban wind simulation. (sources: <https://www.simscale.com/simulations/urban-microclimate/>)

2.1.3. CFD in urban wind simulation

Due to their reduced cost and high physical accuracy, CFD methods have been increasingly employed for urban wind simulations. Several different variations can be applied while perform CFD simulation. Direct numerical simulation (DNS) solves NS equations by explicitly resolving all turbulent scales present in urban environments. The high computational cost makes it computationally prohibitive for large scale simulations, such as urban wind simulation. Consequently, turbulence modeling approaches such as Reynolds averaged Navier Stokes (RANS) and large eddy simulation (LES) are commonly adopted as approximations [82, 21]. RANS computes the time-averaged flow field, and model the effects of turbulent fluctuations through the Reynolds stress tensor. LES applies a spatial filter to the NS equations, explicitly resolving the large eddies while modeling the effects of the small-scale dissipative structures using a subgrid-scale model. The form of incompressible LES can be written as:

$$\begin{aligned} \nabla \cdot \bar{\mathbf{u}} &= 0, \\ \frac{\partial \bar{\mathbf{u}}}{\partial t} + \nabla \cdot (\bar{\mathbf{u}}\bar{\mathbf{u}}) &= -\nabla \cdot \boldsymbol{\tau} - \frac{1}{\rho} \nabla \bar{p} + \nabla \cdot \left(\nu \left(\nabla \bar{\mathbf{u}} + (\nabla \bar{\mathbf{u}})^T \right) \right), \end{aligned} \quad (2.2)$$

where ρ , $\bar{\mathbf{u}}$, and $\bar{p}$ denote the density, spatially filtered velocity and pressure, respectively. ν denotes the kinematic viscosity, and $\boldsymbol{\tau} = \overline{\mathbf{u}\mathbf{u}} - \bar{\mathbf{u}}\bar{\mathbf{u}}$ is the subgrid-scale stress tensor, which must be modeled using an appropriate subgrid-scale model [74].

To avoid explicitly resolving near-wall turbulent structures around buildings and the ground surface, a wall model is employed. In this work, the standard logarithmic law of the wall recommended by the Architectural Institute of Japan (AIJ) guidelines for practical wind environment prediction is adopted [103]:

$$U^+ = \frac{1}{\kappa} \ln(Ey^+) \quad 30 < y^+ < 200, \quad (2.3)$$

where E is the wall roughness constant, $U^+ = \frac{U}{u_\tau}$ is the dimensionless velocity, $y^+ = \frac{yu_\tau}{\nu}$ is the dimensionless wall normal distance. u_τ denotes the friction velocity, and κ is the von Kármán constant.

For the inlet boundary condition, a logarithmic wind profile is adopted to estimate the vertical velocity distribution:

$$\begin{aligned} U(z) &= \frac{u_{ABL}^*}{\kappa} \ln \left(\frac{z + z_0}{z_0} \right), \\ u_{ABL}^* &= \frac{\kappa U_h}{\ln((h + z_0)/z_0)}, \end{aligned} \quad (2.4)$$

where $U(z)$ denotes the inflow velocity at height z , u_{ABL}^* is the atmospheric boundary layer friction velocity, z_0 is the roughness length, and h is the reference height at which the reference velocity U_h is specified. For the other boundaries, symmetry boundary conditions are applied in the spanwise directions, a slip boundary is imposed at the top boundary, and a Neumann boundary condition is used at the outflow boundary.

Currently, in most researches, urban wind simulations rely heavily on general-purpose CFD software packages. While offering high accuracy and versatility, they are often not specifically optimized for the iterative workflows required in early-stage urban design. They typically have steep learning curves and require substantial user expertise to operate [36]. A traditional urban CFD workflow involves a labor-intensive setup stage, including loading building and terrain model, mesh setup, solver setup, boundary setting, simulation domain setting and turbulence setting [22] (Fig. 2.4).

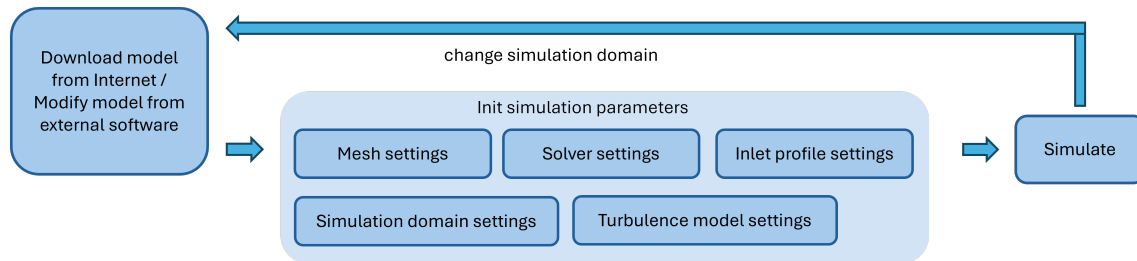


Figure 2.4: A typical urban wind simulation pipeline, taking OpenFOAM [66] as an example.

This complex and fragmented process creates a barrier to utilizing CFD effectively within the fast-paced decision-making processes of urban design [36]. Some plugins have been developed to solve this problem. Butterfly [85] and Eddy3D [47] both provide CFD simulation functions inside Rhinoceros. By applying CFD simulation inside Rhinoceros, they no longer rely on external softwares for building model modification. Moreover, by limiting the research scope into urban wind simulation, they provide preconfigured virtual wind tunnel setups that can be directly used. However, both of them use OpenFOAM as their backend, which runs on CPU and is not efficient enough.

2.1.4. CFD adaptive mesh refinement

In computational wind engineering, artificial boundary conditions are utilized to limit the computational domain size. To minimize the influence of these artificial boundaries on the internal flow field, the computational boundaries must be placed sufficiently far from the region of interest (ROI). Existing best-practice guidelines, including those proposed by AIJ and COST, recommend extending the computational domain beyond the ROI by distances of approximately 5H, 15H, 5H, and 5H for the inflow, outflow, lateral, and upper boundaries, respectively, where H denotes the height of the tallest building within the ROI [103, 27].

Although these recommendations improve simulation reliability, they also result in a substantial increase in computational cost. Assuming that the ROI is a cubic region with side length of H , the resulting computational domain would expand to dimensions of approximately $21H \times 11H \times 6H$. The total simulation volume becomes approximately 1386 times larger, presenting an prohibitively high computational burden.

To alleviate this issue, many urban wind simulation studies employ stretched grid or static local mesh refinement techniques [4, 53, 115, 62]. These approaches allocate different grid resolutions according to the importance of specific regions when the simulation starts. In critical areas, higher resolutions and consequently more accurate simulations are employed to capture important physical phenomenon, while coarser resolutions and simplified simulations are applied in regions of lower importance in order to reduce computational cost.

The partitioning of different resolution levels is primarily determined based on the assumed flow complexity within region. In general, regions closer to obstacles have more complex separated flow structures caused by the interaction between the airflow and buildings, and therefore, required higher spatial resolution. Many studies refine the computational mesh near building surfaces to improve simulation accuracy [95, 62, 53]. Furthermore, some studies also refine the leeward side of buildings in order to capture the strong turbulence structures that frequently occur within building wake regions [4].

Guidelines for mesh generation and refinement strategies have been discussed in the AIJ and COST

best-practice guidelines [103, 27]. These documents provide empirical recommendations regarding maximum mesh resolution, near-building refinement, and the allowable expansion ratio between adjacent grid cells. However, guidance concerning mesh refinement in wake regions have remained undiscussed. As a result, wake-region refinement is not widely adopted in current urban wind simulation practice.

Moreover, despite their widespread use, static mesh refinement approaches present several limitations. First, refinement regions and refinement levels typically need to be manually specified by the user. In particular, the refinement of wake regions is often highly empirical and may require multiple trial-and-error simulations, resulting in significant preprocessing effort and computational overhead [52, 110]. Second, manually predefined refinement regions may either overlook physically important flow structures or allocate excessive computational resources to regions of limited importance [62]. Both cases can negatively affect simulation accuracy and computational efficiency.

AMR criteria for urban wind simulation

These issues represent a general problem in CFD applications, and it is a general practice to employ dynamic adaptive mesh refinement (AMR) techniques [7, 6]. Unlike static refinement methods, dynamic AMR adjusts the mesh resolution during the simulation according to certain criteria. By automatically identifying regions requiring higher resolution, AMR can concentrate computational resources on physically important flow structures while maintaining coarse resolutions elsewhere. Consequently, adaptive refinement has the potential to improve both computational efficiency and simulation accuracy simultaneously.

The utilization of dynamic AMR requires determination of refinement criterias [110]. However, Existing best-practice guidelines for urban wind simulation, including the AIJ and COST guidelines, provide very limited discussion regarding the design of adaptive refinement criteria or other parameters required for adaptive grid generation [103, 27]. The other specialized guidelines focusing on mesh generation also lack exposition on this topic [70]. This absence of standardized guidance may partially explain the limited adoption of AMR techniques in current urban wind simulation research. Existing works include using stream tube and Hessian matrix as refinement criterias, and has only been applied on CPU solvers [37, 113]. The absence of dynamic AMR techniques in current urban wind simulation research might due to the absence of standardized guidance.

GPU AMR algorithms

AMR implementations are commonly categorized into block-structured and octree-based hierarchies [38]. Block-structured AMR partitions the domain into a hierarchy of structured grid blocks. The divided regions are highly-suitable for GPU computation because each block can be processed independently with regular memory access patterns. But the grid subdivision process requires clustering algorithms, which are not straightforward to be conducted fully on the GPU [78]. The additional CPU operations and GPU-CPU memory transfer during AMR refinement is reported to cause significant overhead [112]. In contrast, octree-based AMR can be executed entirely on the GPU. Depending on the refinement granularity, octree-based methods can be further classified into cell and block based solutions. The cell-based solution provides fine grained control over mesh resolution and can achieve a prescribed error tolerance with fewer computational resources [38]. However, its fine level control results in significant data locality issues when implemented in GPU. In contrast, block-based AMR organizes the computational domain into 3D blocks, with each block containing many cells. The data locality between neighbor cells within the same block can thus be preserved.

2.2. Urban noise simulation

In the urban environment, noise propagation is influenced by highly complex physical phenomena, including reflection, diffraction, and refraction caused by wind fields and temperature gradients. Therefore, it is essential to develop accurate and efficient physical models to simulate noise propagation and its interactions with complex urban geometries. Theoretically, the essence of sound is the propagation of density fluctuation within a medium. Thus, the compressible Navier-Stokes (NS) equations can be solved in order to simulate these fluctuations. However, the physical scale of acoustic density fluctuations

is typically several orders of magnitude smaller than that of the bulk velocity field. Consequently, acoustic waves are very easily obscured by the numerical errors of the macroscopic flow, and thus cannot travel over long distances by solving compressible NS equation [18]. Therefore, various computational methodologies have been developed to approximate the noise propagation using different levels of assumptions, including wave-equation-based approaches, geometrical acoustics and energy-based approaches [90].

2.2.1. Wave-based methods

Wave-based methods model acoustics propagation by solving the governing equations of acoustics waves. The most widely adopted governing equation is linearized Euler Equations (LEE). LEE decompose each physical quantity into known background component and acoustics perturbation, $\phi = \bar{\phi} + \phi'$, and then substitute this decomposition into compressible NS equation [5, 104]. The final equation is obtained by neglecting second-order perturbation terms and neglecting the viscous effects.

The LEE is typically solved using finite element (FE) [101], finite-difference time-domain (FDTD) [68] or pseudospectral time-domain (PSTD) methods [34]. In principle, LEE-based modeling is able to provide the most accurate results comparing to the other sound modeling approaches described below. They are therefore widely used in computational aeroacoustics [90, 18]. However, accurately resolving high-frequency acoustic waves imposes strict constraints on spatial grid resolution. For FE and FDTD methods, a element size less than about one-tenth of the shortest acoustics wavelength is generally required [81]. Alternatively, PSTD methods require a maximum grid size smaller than one-half of the wave length to satisfy the Nyquist criterion [34]. These strict resolution limitations make the simulation of medium to high-frequency noise propagation computationally prohibitive, restricting the practical application of LEE solvers in urban sound propagation problems.

2.2.2. Geometrical acoustics

Another group of methods based on geometrical acoustics (GA) are widely used in building-related noise simulation [90]. Instead of solving discretized wave equations, geometric-based methods are used to calculate noise propagation. In GA, noise is assumed to propagate in the form of rays or beams. This assumption neglects the diffraction and interference effects of noise, which becomes significant in low frequency noise. Even though some modelling methods have been proposed to model the diffraction effect [94, 75], GA methods are mostly applied on mid- and high-frequency noise propagation simulations. Different from LEE-based approaches, the computational complexity of GA-based methods mainly depends on the number of sound rays and the geometry complexity of the domain, but not on the domain physical size. It is thus highly suitable for large scene noise propagation simulation, such as urban noise simulation and room acoustics modeling [35, 90].

Deterministic methods

Based on the reflection behavior when a ray hit on a surface, geometrical acoustics methods can be roughly divided into deterministic and stochastic methods. Image source method is a deterministic method, based on the assumption that all the reflections of sound rays are mirror reflections. In order to calculate all the possible reflection ray paths from the source to the receiver, image source method creates many “virtual sources” that contribute to the sound ray paths [90]. Given a noise source and receiver, it works by reflecting the noise source on each reflection boundaries, therefore creating several first-order image sources. Then, each of the first-order image sources reflect on each reflection boundary again, and creating the second-order image sources. This process is repeated until the maximum number of reflections or maximum noise propagation length is reached. Finally, each of the resulting image sources acts like a real sound source, contributing to the final noise level at the receiver.

The complexity of image source method does not depend on the scale of the physical field, which means it can handle very large scene efficiently. However, it has several disadvantages. It assumes all the reflections are mirror reflection, which neglects the diffusive effect of real surface materials, and is not physically accurate [90]. The other problem is it assumes the sound ray propagates in straight line, which neglects the noise propagation refraction due to the existence of temperature difference and wind [10].

Eigen ray tracing method focuses on finding the refracted ray path that connects between the sound source and receiver in non-uniform media [91]. In urban regions, temperature difference and wind can both refract the sound ray. With the existence of the curved travel direction, finding the specific “eigenray” that links between the sound source and receiver becomes difficult. Eigen ray tracing methods usually use adaptive ray direction zooming algorithms to adjust the ray directions [91].

Stochastic methods

Stochastic ray tracing is proposed to accurately simulate reflections from diffuse materials. Instead of explicitly calculating multiple image sources, stochastic ray tracing emits a large number of sound rays with different outgoing directions from the source [90]. When a ray intersects with an interface, instead of reflecting in the specular reflection direction, its reflected direction will be sampled according to some functions, including the physical properties of the surface material using the acoustic bidirectional reflectance distribution function (A-BRDF) [11, 26]. The reflected contribution is then accumulated according to the acoustic rendering equation:

$$L_o(\mathbf{x}, \omega, t) = L_e(\mathbf{x}, \omega, t) + \int_{S^2} f(\mathbf{x}, \omega, \omega_i) L_i(\mathbf{x}, \omega_i, t) \cos(\theta_i) d\omega_i, \quad (2.5)$$

where $L_o(\mathbf{x}, \omega, t)$ denotes the outgoing acoustic energy at space position $\mathbf{x}$ in direction ω at time t . Similarly, $L_e(\mathbf{x}, \omega, t)$ and $L_i(\mathbf{x}, \omega_i, t)$ denote the emitted and incident acoustic energy, respectively. ω_i represents the incident direction. Inside the integral, $f(\mathbf{x}, \omega, \omega_i)$ is the A-BRDF, and θ_i is the angle between incident direction and surface normal.

Comparing to deterministic methods, stochastic ray tracing provides a physics-based model of acoustic reflection. With a sufficiently large number of rays, stochastic ray tracing can accurately capture the propagation behavior of high-frequency sound waves.

2.2.3. Energy-based methods

Instead of solving the propagation of wave, energy-based methods model the transport of sound energy. The strict resolution requirement is therefore greatly relaxed, allowing simulations to be performed on much coarser grids. One of the most widely used energy-based methods is the diffusion model. It is based on the assumption that the spatial distribution of sound energy density satisfy a diffusion equation, similar to heat diffusion [71]. Due to the coarser resolution used, simulating sound propagation using diffusion model is much cheaper than wave-based methods. However, the diffusion assumption is only valid when the sound field has been fully developed. Thus, it cannot handle the cases when the receiver is close to the sound source or in open spaces. Its applicability is therefore limited.

2.2.4. Noise propagation in urban region

The Environmental Noise Directive (END) 2002/49/EC, adopted in 2002, requires EU member states to assess and publicly report environmental noise levels in urban areas, as illustrated in Fig. 2.5. Accurate noise propagation models are therefore essential for estimating sound levels at different locations within an urban region. Multiple noise propagation models have been proposed [87, 24], the current common assessment method in Europe is CNOSSOS-EU [48].

In CNOSSOS-EU, as well as in many other urban noise prediction frameworks, sound propagation is modeled using a semi-empirical engineering model largely based on geometric acoustics. This approach combines the image source method to model reflections from the ground and building surfaces, arc shaped ray tracing to account for refraction due to non-uniform air density, and convex hull-based diffraction models to represent diffraction across building edges. These methods are computationally efficient and well suited for generating noise maps. However, the combination of multiple approximations limits their accuracy in acoustically complex environments.

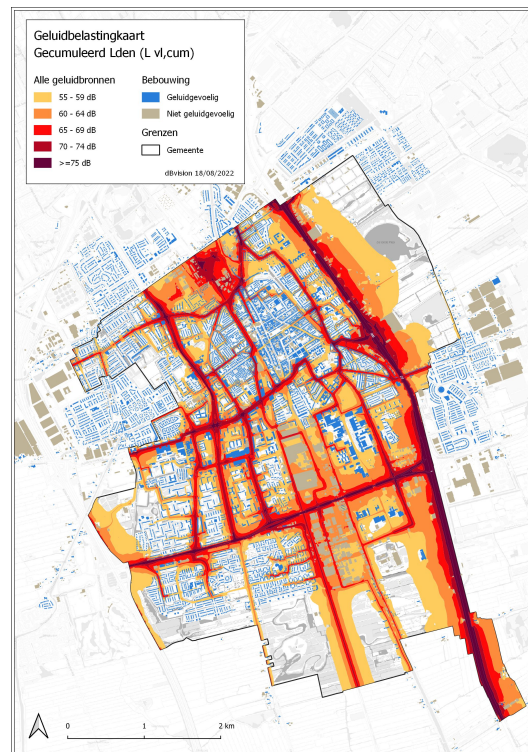


Figure 2.5: The noise map of Delft, Netherlands. (Source: <https://www.delft.nl/lucht-en-geluidskwaliteit>)

2.2.5. Sound ray refraction models

Due to their high computational efficiency in large-scale scenes and their relatively high physical accuracy, ray tracing-based sound propagation simulation methods have been widely adopted in acoustics modeling, including numerous applications within urban spaces [111, 59, 71, 77].

One of the primary factors influencing both the accuracy and efficiency of urban noise propagation simulation is the treatment of sound ray refraction. The direction of sound propagation is strongly affected by the surrounding atmospheric medium. In urban environments, sound often undergoes refraction during propagation due to wind and temperature variations in the air. Existing approaches for modeling sound ray propagation can generally be categorized into three groups. The first category assumes that sound rays propagate along straight-line trajectories [116, 117]. This method is computationally efficient and straightforward to implement. However, it is only applicable under windless and homogeneous air conditions where refraction effects can be neglected. The second category models sound ray trajectories as circular arcs, where the curvature is determined by the temperature gradient between different altitudes in the environment [79, 87, 48]. These approaches can partially account for atmospheric refraction while maintaining relatively low computational cost. Nevertheless, the modeling process typically assumes spatially uniform temperature gradients, resulting in a lack of precision. The third category discretizes sound propagation paths into multiple segments, where the propagation direction within each segment is determined by local wind velocity and temperature gradients [58, 59, 14]. These methods can achieve substantially higher physically accurate sound propagation paths. However, the determination of ray paths requires numerically solving the ray equations or performing geometric approximations for each segment during ray propagation, resulting in significantly higher computational costs than the previous approaches.

Another critical factor affecting the computational efficiency of sound ray tracing simulation is the number of emitted rays. In order to reduce the variance of ray tracing sampling, as well as to generate noise map, a large number of rays must be emitted in various directions, while their reflections within the environment must be evaluated individually. This workload is highly suitable for GPU acceleration. However, although several studies have explored GPU-based acceleration for urban acoustics ray tracing, existing implementations have primarily focused on cases that neglect refraction effects [20, 117]. The

segment-based refraction aware noise propagation model, which has higher physical fidelity and also involve higher computational complexity, have received limited attention. Furthermore, existing GPU-accelerated ray tracing urban noise simulation methods solely utilize general purpose GPU computing for acceleration, neglecting the use of dedicated hardware ray tracing units. Recent studies have demonstrated that specialized ray tracing hardware can sometimes yield performance improvements hundreds of times greater than traditional streaming multiprocessors in certain scientific computing applications [57]. Ray tracing shaders inside game engines can provide access to the ray tracing hardware, it is thus worth exploring to use them in urban noise simulation.

2.2.6. Ray trace sampling strategies

Importance sampling

Besides the refraction treatment, another factor that significantly affects the computational efficiency and convergence rate of sound ray tracing is the sampling strategy during reflection.

When a sound ray reaches an interface, the distribution of reflection energy is determined by the material properties of the surface. The function used to describe this energy distribution is often referred to as acoustics-bidirection reflection distribution function (A-BRDF) [11]. Most urban buildings and terrain have diffuse surface, which means an incoming sound ray will be scattered over a large range of directions, each carrying a portion of the reflected energy.

Ideally, the reflected energy should be evaluated by integrating the contributions over all possible reflection directions. However, modelling this continuous reflection process exactly is computationally prohibitive in practical applications. Instead, only a limited number of reflection directions can be sampled and traced further. The reflected contribution is therefore approximated using Monte Carlo integration:

$$F = \frac{1}{N} \sum_{i=1}^N \frac{f(X_i)}{p(X_i)} \quad (2.6)$$

where F is the Monte Carlo estimator, N is the number of samples, $f(X_i)$ denotes the i_{th} sampled value, $p(X_i)$ is the probability density function (PDF) used for sampling.

The choice of sampling PDF can lead to different estimation variance. It can be proved that the estimation with minimum variance is obtained when:

$$p(x) \propto |f(x)|, \quad (2.7)$$

which indicates that the sampling PDF should be as similar as the function being integrated as possible. The integrand in the rendering equation Eq. 2.5 can be written as

$$f(x) = f(\mathbf{x}, \omega, \omega_i) L_i(\mathbf{x}, \omega_i, t) \cos(\theta_i), \quad (2.8)$$

where $f(\mathbf{x}, \omega, \omega_i)$ and $\cos(\theta_i)$ can be calculated, but the incident sound energy L_i is unknown before tracing the sampled ray. Different sampling strategies can therefore be categorized according to the approximations they employ for the integrand.

The commonly used importance sampling (IS) strategy samples according to the A-BRDF and cosine term, i.e., $f(\mathbf{x}, \omega, \omega_i) \cos(\theta_i)$, while neglecting the sound energy term L_i . Related techniques are widely used in acoustics propagation simulations due to their simplicity and efficiency.

However, IS strategy shows limitations in certain scenarios. In Fig. 2.6 (a), a sound ray is shoot from the receiver (circle) to a rough surface. A (reversed) reflection ray is then sampled from the intersection point. If the reflection ray is sampled according to A-BRDF, there is a high probability that the sampled ray will fail to reach the sound source (square). In this case, this contribution from the source will be neglected, leading to an under estimated sound energy level.

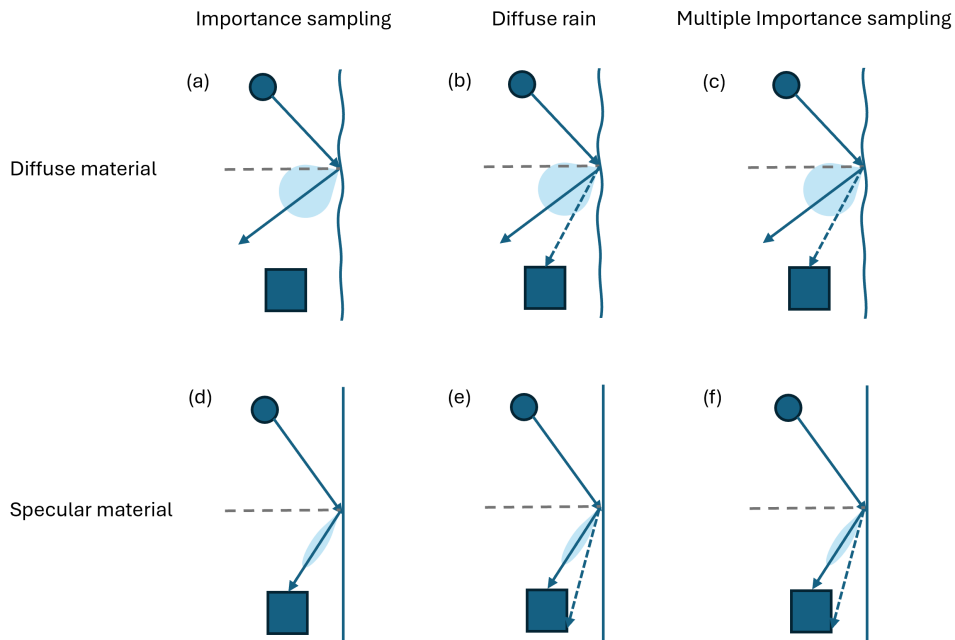


Figure 2.6: Comparison between three sampling strategies on two different materials. The circle is sound receiver, the square is sound source. The solid line arrow represents the inlet sound ray and reflected sound ray using A-BRDF sampling, the dotted line arrow represents the ray obtained from noise source sampling.

Diffuse rain

The under estimation of sound energy level in conventional IS results from neglecting the contribution of incident sound energy term L_i , in Eq. 2.8. A solution to this problem is the diffuse rain algorithm [69, 93], which can be regarded as an acoustic variant of next event estimation (NEE) in computer graphics [49].

The key idea of diffuse rain is to explicitly sample regions associated with high L_i values, which generally correspond to high incident sound energy. When a sound ray interacts with a surface, two reflected rays are generated. The first ray is sampled using the conventional IS strategy, and is responsible for estimating the indirect sound contribution. This ray is recursively traced through the scene. Moreover, another secondary ray is generated by directly sampling the sound source and connecting the sampled point to the reflection point. If the connection is not occluded by obstacles, the direct contribution from the sound source is accumulated into the final result, as illustrated in Fig. 2.6 (b). Since the direct sound contribution has already been explicitly evaluated by source sampling, contribution from sound source reached by the A-BRDF sampling ray in this round of reflection calculation is ignored to avoid double counting and preserve energy conservation.

Diffuse rain algorithm achieves better performance than traditional IS in most diffusive material scenarios. However, it also exhibits limitations in certain cases. As illustrated in Fig. 2.6 (e), when the surface material becomes highly specular, the A-BRDF support region becomes extremely narrow, meaning that the reflected energy is concentrated within a very small range of directions. In such cases, conventional IS becomes more suitable, as shown in Fig. 2.6 (d). If the diffuse rain algorithm is applied instead, as shown in Fig. 2.6 (e), the directly sampled source ray has a high probability of falling outside the specular BRDF lobe, resulting in only a negligible contribution to the final estimation. Although the first ray may still reach the sound source, its contribution cannot be accumulated simultaneously due to energy conservation requirement. Consequently, specular reflections will be underestimated in certain cases, leading to an underestimated sound energy level.

Multiple importance sampling in computer graphics

The rendering counterpart of the above problem, the NEE, has been widely studied in the field of computer graphics. One of the proposed solutions is multiple importance sampling (MIS), which combines different sampling strategies into a single estimator [109].

Different from diffuse rain, MIS combines samples generated by all strategies and weights their contributions according to their sampling probabilities. The core MIS estimator is given by

$$F = \sum_{i=1}^n \frac{1}{N_i} \sum_{j=1}^{N_i} w_i(X_{i,j}) \frac{f(X_{i,j})}{p_i(X_{i,j})}, \quad (2.9)$$

where n is the number of sampling strategies, N_i is the number of samples generated by the i_{th} strategy, $p_i(X_{i,j})$ is the corresponding sampling PDF of the current sample $X_{i,j}$, and w_i denotes the MIS weighting function. f is the integrand. Different definitions of the weighting function have been proposed, the original MIS paper introduces the balance heuristic:

$$w_i(X) = \frac{N_i p_i(X)}{\sum_{k=1}^n N_k p_k(X)}. \quad (2.10)$$

MIS addresses the limitations of both IS and diffuse rain sampling by adaptively weighting different sampling strategies according to their effectiveness. As illustrated in Fig. 2.6 (c), when the material is diffuse, directions sampled from the sound source are generally associated with much higher source-sampling PDFs than IS PDFs. Consequently, the contribution from direct source sampling receives a larger MIS weight, and the estimator behaves similarly to next event estimation. In contrast, As illustrated in Fig. 2.6 (f), when the material is highly specular, the IS PDF becomes significantly larger around the specular reflection direction, while the contribution of direct source sampling remains similar. The MIS estimator will thus behave similarly to conventional IS. Therefore, MIS can adaptively combine the advantages of both sampling strategies and achieve robust performance across both diffuse and specular reflection scenarios.

2.3. Game engine

Game engines are software frameworks originally designed for video game development [107]. Alongside the evolution of modern games, game engine technologies have advanced significantly in their functionality and computational capabilities. Beyond game development, these frameworks are also increasingly employed in general software engineering, particularly within scientific software development. They have been utilized in molecular viewer and simulation [55], the training and validation of autonomous driving systems [23], and the visualization of simulated wind fields [8]. Their widespread adoption in scientific software development stems from several inherent advantages, including versatile user interaction frameworks, integrated visualization systems and high-fidelity rendering quality. Despite from these advantages, its capacity for massively parallel computation and efficient ray-mesh intersection also makes it well-suited for computationally intense tasks. Four well-developed techniques in modern game engines are particular relevant to this project:

- Massively parallel computation
- High performance ray-mesh intersection
- Intuitive user interaction
- Efficient visualization

2.3.1. Parallel computation

In order to simulate virtual environments, millions of pixels needs to be rendered in each frame. The complex material systems used employed by some games further increase the needed computational load. To achieve high frame rates under these conditions, Graphics Processing Units (GPUs) were developed to accelerate highly parallelized operations [96]. Game engines natively support GPU acceleration through programming models known as shaders. In response to the growing demand for GPU-accelerated general-purpose computing, compute shaders were introduced to execute non-graphical computations. By utilizing these compute shaders, modern game engines can function similarly to dedicated parallel

programming frameworks, such as CUDA [64] and ROCm [2].

2.3.2. Ray-mesh intersection

Ray tracing is a technique that can enhance the render visual fidelity under complex lighting conditions. The most time consuming operation during ray tracing is the intersection detection between ray and objects in the scene. Modern GPUs utilize specific hardwares (ray tracing cores) to accelerate this process [114]. The corresponding programming model in game engines is named ray tracing shaders. Beyond optical rendering, ray tracing shaders can also be repurposed for other ray-mesh intersection detection tasks, which can facilitate our project in the noise simulation part. The architecture of a typical modern GPU, including the modules to accelerate parallel computing and ray-mesh intersection is shown in Fig. 2.7.

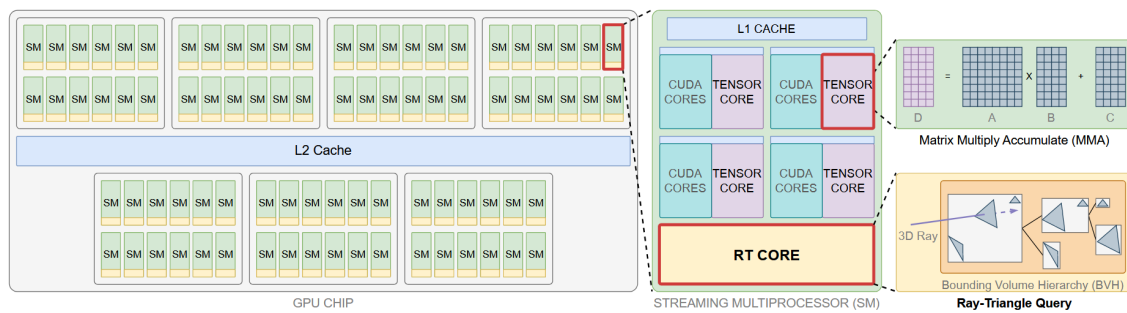


Figure 2.7: Visualization of typical modern Nvidia GPU architecture. The CUDA cores are used for general-purpose parallel computation acceleration, while the RT cores are specifically designed for ray-triangle intersection calculation. (Source: [57])

2.3.3. User interaction

In order to facilitate the game development process, game engines provide robust frameworks for runtime user interaction. They often provide CPU-based ray-object intersection system to detect and select entities within the 3D environment. Furthermore, they offer mature mechanisms for the dynamic creation, scaling, moving (translation and rotation), and deletion of objects. These platforms also provide user interface (UI) systems that can map user inputs to underlying programmatic behaviors. In the context of environmental modeling, these capabilities facilitate the real-time manipulation of urban morphology and other user interactions.

2.3.4. Visualization

Game engines are renowned for their high-fidelity visual rendering, driven by powerful built-in visualization systems. For example, both Unity's visual effect system (VFX) and Unreal Engine's Niagara system can generate and manipulate millions of particles simultaneously on the GPU [106, 25]. Visualization is also an essential part in wind and noise simulation. By leveraging these advanced particle systems and rendering pipelines, simulated volumetric data can be visualized dynamically. This advantage can enhance the interpretability of simulation outputs, providing researchers and practitioners with intuitive and real-time insights into the wind and noise behavior.

Virtual reality

Beyond conventional screen-based visualization, virtual reality (VR) can further enhance the interpretation of urban wind and noise simulation results. These techniques can provide immersive and spatially situated views of wind and noise data, allowing users to perceive wind and noise information in real pedestrian locations.

In urban wind simulation, VR can help users visualize complex airflow phenomena such as downwash wind, wake structures and channeling effects by placing the observer directly inside the simulated environment. Particle data and streamlines can also be explored interactively from different viewpoints. Similarly, in urban noise simulation, the noise level can be directly mapped into the urban region, enabling users to directly feel it while walking in the virtual street. These enhancements can significantly increase

the interactivity and immersion level of the proposed solver.

2.4. Geographical information system

Geographical Information Systems (GIS) serve as foundational frameworks for storing, managing, analyzing, editing, visualizing, and exporting spatial and geographic data [16]. A GIS acts as a dynamic spatial database that links geographic locations with descriptive attributes and environmental information.

2.4.1. Use GIS for urban wind and noise simulation

Accurate urban wind and noise simulation requires high-fidelity real-world data to establish correct boundary conditions. One of the key capabilities of GIS is the acquisition and analysis of spatial data from real-world environments. Their integrated ability of connecting to heterogeneous data sources and directly utilize spatial datasets makes GIS frameworks highly suitable for urban planning and environmental analysis. Related urban wind and noise simulation data sources that can be directly integrated into GIS workflows include:

- **Topographical data:** Topographical data including terrain data, building information and tree data is important in urban wind and noise simulation. Platforms such as OpenStreetMap [67] and 3DBAG [73] provide extensive urban data, including building footprints and height information. OpenStreetMap also provides landuse information which can be utilized to determine the acoustics behavior in these regions. Open data platform in each municipality provides data about tree location and basic informations [28, 31]. In addition, datasets such as AHN5 [1] provide high-resolution point cloud data for terrain elevation and vegetation geometry within the Netherlands.
- **Meteorological data:** To reproduce realistic environmental conditions, climatological data from national meteorological institutions are required. KNMI [86] in the Netherlands can provide data include wind speed, wind direction and temperature profiles, which acts as important inputs for CFD and acoustics applications.
- **Noise source data:** Major urban noise sources include road traffic, railway systems and aircraft activities. Platforms such as Nationaal Dataportaal Wegverkeer (NDW) and Nationaal Wegen Bestand (NWB) provide road network and traffic-related information [61, 83], which can be used in noise source identification. In addition, Prorail provides railway-related noise data [76], while Luchtverkeersleiding Nederland (LVNL) provides aircraft noise and air traffic data [54], enabling the incorporation of railway and aviation noise sources into urban noise simulations.

2.4.2. Leveraging Geospatial Database Management Systems for visualization

Geospatial Database Management System (Geo-DBMS) are widely used for storing, indexing, and querying large-scale spatial datasets efficiently [3]. They enable scalable management of complex urban simulation data by providing specialized spatial acceleration structures and efficient query mechanisms.

In our project, a Geo-DBMS can serve as the central repository for the simulated results. Spatial data including voxelized fluid field generated by the wind solver, noise map and curve based acoustics propagation paths generated by the noise solver can be saved into the Geo-DBMS. When the user queries the simulation data at various geographical locations, related simulation results can be efficiently retrieved by the Geo-DBMS and then forwarded to the renderer for visualization.

2.4.3. Limitations of traditional GIS frameworks

Despite their strong capabilities in spatial data management and analysis, traditional GIS frameworks generally lack the ability to perform physical simulations directly. They cannot solve Navier-Stokes equation based fluid simulation for urban wind analysis, nor can they perform wave equation-based or ray tracing-based acoustics simulations. This limitation restricts the direct application of traditional GIS platforms in wind and noise-aware urban planning workflows.

2.4.4. Typical GIS-simulation interoperability pipeline

Currently, the integration of GIS with wind and noise simulation relies on a fragmented, multi-step pipeline. A typical workflow involves fetching data from related datasets, clipping the data utilizing GIS softwares or geometric processing softwares, potential format conversion and exportation, simulation, and post-analysis.

Although functional, this workflow is often time-consuming and lacks interactivity. The manual transfer of data between different software environments introduces significant inefficiencies and creates bottlenecks in the simulation pipeline. Furthermore, the absence of a real-time, bidirectional connection between GIS platforms and physical solvers limits iterative design exploration and interactive urban analysis. These limitations highlight the need for integrated and real-time simulation frameworks for urban wind and noise analysis.

3

Methodology

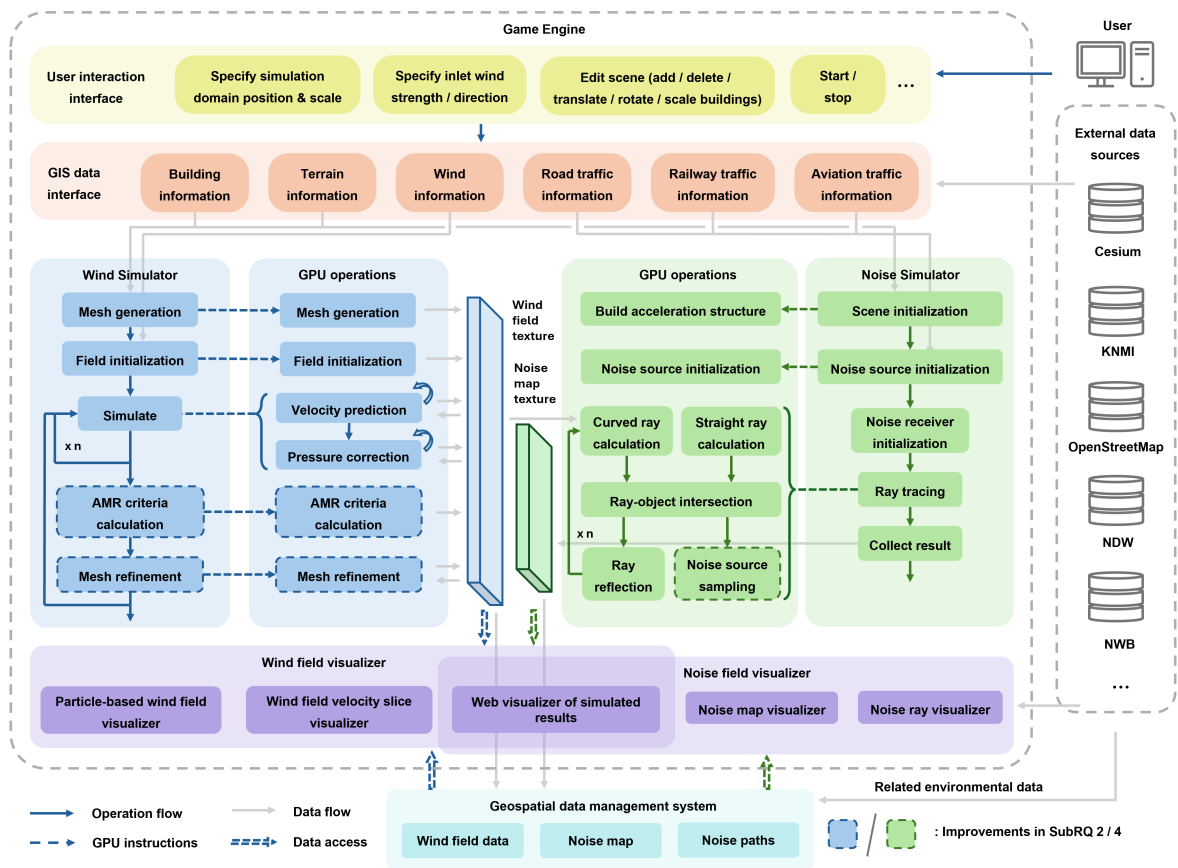


Figure 3.1: The outline of our proposed urban wind and noise solver. The core of this solver is an urban wind simulator (blue) and an urban noise solver (green), both implemented in the Unity game engine and accelerated using GPU. The solvers are controlled by a user interaction interface (yellow), receive input from external data sources (right dotted rectangle) through an integrated data interface layer (orange), and visualize the results through a visualization system (purple). A DBMS system (aquamarine) is attached to save and index the simulated results for fast retrieval and future visualization.

The goal of this project is to develop an interactive and computationally efficient urban wind and noise simulation framework, as illustrated in Fig. 3.1. The four proposed research subquestions will be addressed in a progressive sequence.

First, the research focuses on utilizing the high-performance computing capabilities of game engine to implement an urban wind solver. The built-in visualization and user interaction functionalities of the

game engine will also be employed to enhance the usability and interactivity. In addition, the solver will integrate necessary GIS functionalities to enable direct interaction with real-world spatial datasets. This stage aims to establish an urban wind simulation framework with hardware-accelerated performance and outstanding interactive capabilities, as illustrated in the left part of Fig. 3.1.

Second, the research still focus on the left part of Fig. 3.1, investigating algorithmic approaches for further improving computational performance. In particular, the effectiveness of adaptive multi-resolution methods in urban wind simulation will be explored. Upon completion of this stage, the project aims to produce an urban wind simulator with enhanced performance and urban wind specific adaptive multi-resolution techniques.

Following this, the author will build an urban noise simulator that couples with the urban wind simulator, as illustrated in Fig. 3.1. This phase first focuses on using modern compute shader to accelerate refraction-aware urban noise simulator. It will also explore the possibility of using ray tracing shaders in urban noise simulator. Then, after the simulator has been built, we will focus on using realistic wind simulation data to improve the physical fidelity of acoustics propagation simulation, particularly for modeling sound refraction caused by wind fields and temperature gradients. The goal of this stage is to develop a computationally efficient and physically accurate urban noise simulator that can be seamlessly integrated with the wind simulation framework.

Finally, the research focuses on how to reduce the ray tracing sampling variance by improving the sampling schemes, as illustrated in Fig. 3.1. The author plans to adapt multiple importance sampling, a technique that can balance between BRDF sampling and source sampling into urban acoustics simulation. This technique can, in theory, solve the problem introduced by ordinary importance sampling and diffuse rain. This research will then focus on how to adapt this technique into refraction-aware urban noise simulation.

3.1. Improving usability and efficiency of wind simulator through game engine and GIS integration

As mentioned in Sec. 2.1.3, current CFD software and plugins face problems in either urban interactivity and efficiency. In this research subsection, we explore the SubRQ 1:

“How to utilize a game engine-GIS-CFD triplet to improve the computational efficiency and usability of urban wind simulation?”

Specifically, we propose a coupled framework that integrates CFD solver and some GIS functionalities inside game engine, and interacts with the outside data sources to obtain real world geographical data. An overview of the proposed framework is shown in Fig. 3.1.

3.1.1. Coupling CFD with a game engine

Game engines are primarily designed to support interactive and real-time digital environments. As a result, they provide integrated functionalities for user interaction, high-performance computation, and efficient visualization. These features align well with the requirements of an urban wind simulation system.

Several game engines are currently widely adopted in industry and research, including Unity, Unreal Engine, and Godot. Among these alternatives, Unity is selected as the primary development platform for this research. Unity offers extensive documentation, a large developer community, strong cross-platform compatibility, and a relatively lightweight development workflow based on the C# programming language. Although Unreal Engine generally provides higher rendering fidelity, Unity is considered more suitable for rapid prototyping and scientific computing integration due to its multi-platform support and more lightweight programming framework.

Improving computational efficiency using game engine

Modern game engines inherently support GPU-based parallel computation. In Unity, compute shaders provide a framework for implementing general-purpose, massively parallel numerical algorithms, making them particularly suitable for accelerating computational fluid dynamics (CFD) simulations. Fig. 3.2 illustrates the computational pipeline of a typical CFD solver based on the SIMPLE [72] algorithm, implemented on both CPU and GPU architectures. In the GPU-based pipeline, all the heavy numerical computations are offloaded to the GPU through compute shader kernels, while the CPU is only responsible for kernel dispatching and convergence monitoring. The computational efficiency can thus be significantly improved due to the high computational throughput of modern GPU.

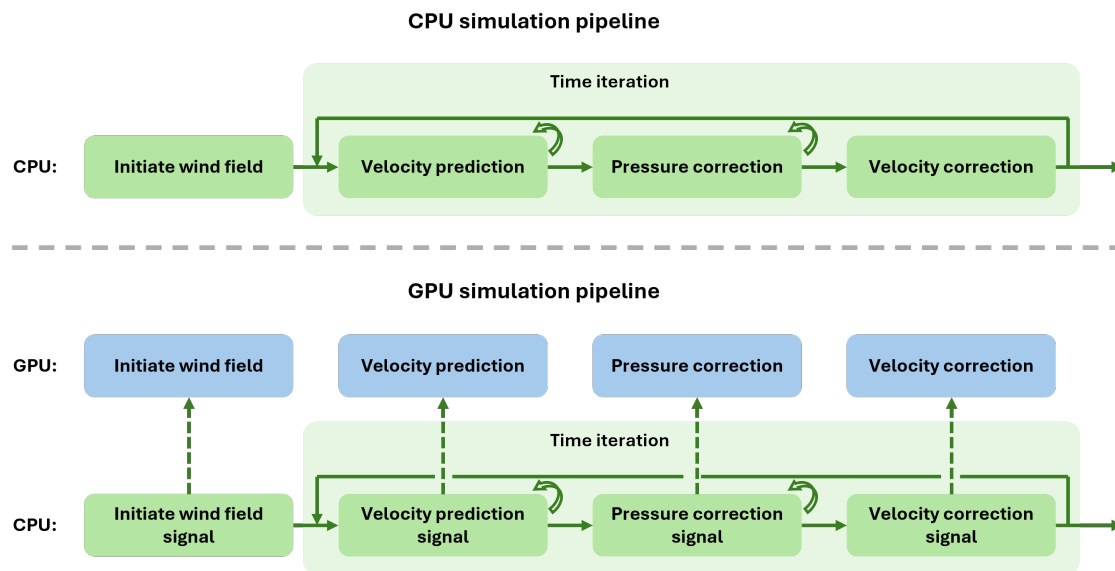


Figure 3.2: A comparison between CPU and GPU CFD simulation pipeline. In a typical GPU CFD simulation, basically all the computation is offloaded to the GPU to utilize its powerful parallel processing ability.

In addition to GPU acceleration, Unity and other game engines provide built-in ray casting systems for efficient geometric queries. These systems internally employ spatial acceleration structures, such as bounding volume hierarchy (BVH) [63], to accelerate ray-object intersection tests. The intersection queries can thus be performed within logarithmic computational complexity. In this project, the ray-casting function is utilized during the voxelization stage of the wind field, determining whether computational cells are located inside or outside the fluid region.

Enhancing interactivity using game engine

Besides shader-based acceleration, game engines also provide extensive built-in functionalities to support user interaction during runtime, which are highly beneficial for interactive urban wind simulation systems.

In Unity, the uGUI system enables the efficient construction of graphical user interfaces (GUIs), including text displays, input fields, sliders, and interactive buttons [105]. Unity's event-driven architecture enables real-time interaction between user interactions and simulation logic. Specifically, the Unity Event System detects user actions such as text input, button clicks, and slider adjustments. It subsequently triggers the corresponding functions defined in the simulator. This mechanism allows users to dynamically modify simulation parameters and interact with the computational environment during runtime.

Furthermore, Unity's built-in ray-casting functions can also be used to facilitate interaction with objects in three-dimensional space. By shooting a ray from the corresponding screen pixel, intersection tests can be performed to determine the selected physical location within the environment. This functionality is particularly important for interactive scene construction during simulation.

Improving visualization using game engine

Modern game engines provide highly optimized rendering and visualization pipelines that can be effectively utilized for scientific visualization tasks. In the proposed framework, the urban environment, together with simulated wind fields, is visualized using the real-time rendering capabilities of the game engine.

A key advantage of this design is that both the CFD simulation and the visualization pipeline are executed directly on the GPU. Consequently, the visualization process can directly access simulation data stored in GPU memory without requiring additional CPU-GPU data transfers. This zero-copy visualization strategy significantly reduces communication overhead and enables efficient real-time rendering of wind field simulation results.

Based on this GPU-driven visualization framework, multiple visualization techniques can be employed to represent different characteristics of urban wind. For two-dimensional analysis, mesh-based rendering can be used to visualize wind field slices. For three-dimensional analysis, particle-based visualization implemented through Unity's VFX system can illustrate complex flow structures. In addition, direct rendering of geometric primitives can be utilized to generate streamlines and other glyphs, providing intuitive representation of wind direction.

The visualization system provided by game engines improves the interactivity and visual quality of the simulation, which is particularly important for urban wind related decision making process.

3.1.2. Enhancing usability by coupling with GIS functionalities

Although modern game engines provide strong capabilities for parallel computation, user interaction, and real-time rendering to support simulation related applications, they are primarily designed for virtual environments. To enhance the practical usability of the proposed urban wind simulation system, it is essential to integrate real-world geographic information into the simulation workflow. This objective can be achieved by incorporating selected GIS functionalities into the game engine environment.

Through GIS integration, large-scale urban geometry data, including terrain and building information, can be imported directly into the simulation system. This enables the rapid construction of urban simulation domains based on real-world environments. Compared with manually constructing simulation scenes, such approach can significantly reduce the human effort for scene preparation and improve the scalability of the system for large urban areas.

Moreover, real-world meteorological data can also be incorporated into the simulation framework. Environmental measurements such as wind speed and wind direction can be utilized to define inflow boundary conditions for the CFD simulation. As a result, the simulator can operate under realistic environmental conditions and better reflect real-world urban wind behavior.

The implementation of this architecture in Unity has already been completed as preliminary work. This includes the integration of different geographic data sources, the validation and performance evaluation of the solver. The details will be presented in Sec. 4.

Efficient saving and retrieving of the simulated wind data is also very important for future visualization and analysis. Geo-DBMS can be integrated into the system to allow for efficient data access. Using the high performance geo indexing data structures, the simulation results saved into Geo-DBMS can be very efficiently retrieved and visualized.

3.2. Improving efficiency of wind simulator through adaptive mesh refinement

As discussed in Sec. 2.1.4, existing urban wind field simulation often use static mesh refinement rather than dynamic adaptive mesh refinement. However, static mesh refinement may lead to additional labour in mesh division, computational resource waste, and reduced simulation accuracy in certain regions [70]. This situation may result from the lack of systematic guidance for adaptive mesh refinement (AMR) in

the field of urban wind simulation. This subsection aims to develop an AMR framework using Unity compute shader, and explore the suitable refinement criteria that achieves the best accuracy and efficiency for urban wind simulation. This subsection corresponds to subRQ 2:

“To what extent can the computational efficiency of urban wind simulator be improved through adaptive mesh refinement?”

The overall workflow is illustrated in Fig. 3.3 and consists of the following key steps:

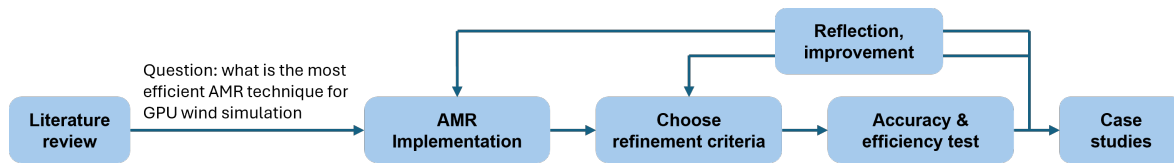


Figure 3.3: The research pipeline of sub-question 2.

3.2.1. Implementing AMR algorithm in compute shader

The first step in this SubRQ is to develop an AMR algorithm suitable for compute shaders. In order to maximize the GPU throughput and therefore computational efficiency, the author aims to apply the AMR algorithm fully on GPU. As discussed in Sec. 2.1.4, most AMR algorithms have efficiency problems on GPU. Block-structured AMR cannot be fully implemented on GPU, while point-based octree AMR faces data locality issues, limiting its efficiency on GPU. Due to its benefits in GPU efficiency and enhanced memory access consistency, block-based octree AMR has been selected as the core data structure for the urban wind simulator [38]. The computational grid will be partitioned into blocks containing dozens of cells, and grid refinement will be applied to each block. By adopting this paradigm, grid points within the same block can identify neighboring points directly via index offsets, therefore preserving data locality within each block. Tree-based data queries only need to be performed at the interfaces between blocks. Therefore, substantial computational resources can be conserved, and the parallel computing throughput can be maximized.

3.2.2. Develop grid refinement criteria

Since the best practice guidelines in urban wind simulation did not provide specific grid refinement criteria for AMR, it is an important practice to design a robust set of criteria 2.1.4. These criteria will be programmed as trigger conditions in the compute shader to automatically evaluate whether a block should be split or merged, and will be executed each several time steps. Three main categories of criteria are planned to be considered, developed and tested:

- Geometric criteria: distance to the buildings and ground surfaces. Shorter distances to physical boundaries require smaller grid sizes to accurately capture the boundary layer flow and the interaction between wind and urban structures. The design of this criteria will refer to the guidelines of static mesh refinement.
- Error / physical properties criteria: error in the flow field, represented by Hessian matrix, or intensity of turbulence within the flow field. Regions exhibiting high error that cannot be represented by current mesh resolution will be refined. Regions with highly dynamic flow behaviors, such as strong turbulence in building wake regions, will also dynamically trigger increased resolution.
- Region of Interest (ROI) criteria: specific, user defined areas within the simulated flow field, such as a certain pedestrian-height zones, are of higher importance for urban comfort assessment and will be assigned higher weighting factors. Algorithms such as windward stream tube [37] can also be incorporated to refine the flow regions that make the largest contribution to the ROI.

3.2.3. Validation and testing

Finally, the accuracy and computational efficiency of the developed AMR framework will be tested, followed by several urban case studies. The performance of the AMR solver will be benchmarked against

our solver developed in Sec. 3.1.1, as well as static mesh refinement techniques. By evaluating how different sets of refinement criteria impact the simulation accuracy and computational cost, the ultimate goal of this stage is to formulate a set of best-practice guidelines for applying AMR in urban wind simulations. This section aims to address the gap in the AIJ and COST guidelines, providing valuable insights for future wind modelling researches and practices.

3.3. Improving accuracy and efficiency of noise simulator using wind simulator and game engine

As discussed in Sec. 2, among existing ray tracing-based urban noise propagation methods, stochastic ray tracing combined with segmented refraction modelling provides one of the highest level of physical fidelity, but is also computationally expensive. Both the large amount number of rays introduced by stochastic ray tracing, and the heavy and sequential computation of segmented ray tracing can significantly slow down the computation. This section aims to address the issue using hardware acceleration provided by modern game engines. This section also aims to coupling the ray tracing noise simulation with the wind field simulator developed in Sec. 3.1. This subsection corresponds to subRQ 3:

“How can an integrated framework combining game engine, GIS, CFD and CA optimize the physical fidelity, computational efficiency and usability of ray tracing-based urban noise simulator?”

The problems to be resolved in this section are divided into two steps. The first stage focuses on the development of a GPU-accelerated urban noise simulation framework within the Unity game engine. In particular, this stage aims to investigate the use of compute shaders for accelerating segment-based sound propagation methods, and the potential use of hardware-accelerated ray tracing shaders for straight line propagations. The possibility of combining compute shaders and hardware ray tracing techniques to further improve computational performance will also be investigated. The outcome of this stage will be a highly efficient urban acoustic simulation framework capable of handling large-scale environments. The second stage focuses on coupling the acoustics solver with the urban wind simulation framework developed in the preceding chapters. This stage aims to evaluate the feasibility and limitations of existing sound refraction algorithms when applied to realistic wind field, and proposing corresponding improvements. The outcome of this stage will be a highly efficient urban wind-noise simulation framework. An overview of this pipeline is demonstrated in Fig. 3.4.

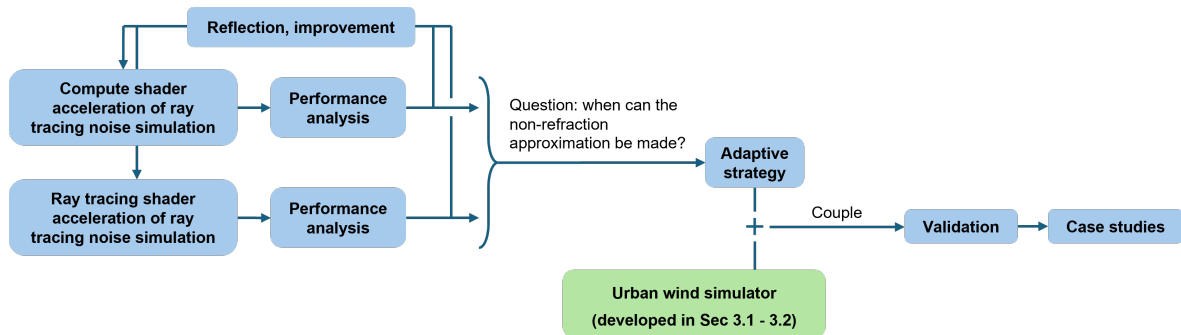


Figure 3.4: The research pipeline of sub-question 3.

3.3.1. Accelerating ray tracing calculations with game engine

Existing ray tracing studies for urban noise simulation primarily focus on accelerating straight-line ray tracing. Under such assumptions, the dominant computational cost arises from the intersection tests between each sound ray and the surrounding scene geometry. However, in physically more accurate refraction-aware models, each sound ray undergoes multiple times of refraction according to the atmospheric condition, and intersection tests must be performed repeatedly during each propagation path. These additional computations significantly increase the overall computational cost of the simulation. In this subsection, the author aims to investigate how modern compute shaders can be utilized to accelerate

refraction-aware acoustic ray tracing methods for urban noise simulation.

Simultaneously, the use of modern ray tracing shaders to accelerate noise simulation that neglect refraction will be investigated. A straight-line ray tracing solver will be implemented, and the applicability of this approximation under different atmospheric conditions (low wind velocities or minimal temperature gradients) will be evaluated. If possible, the project aims to develop a hybrid acoustic simulation framework that dynamically switch between different propagation models according to the physical conditions and accuracy requirements of the simulation. The solver will employ straight-line ray tracing to improve computational performance when the approximation conditions are satisfied. Conversely, when atmospheric refraction becomes significant or higher physical accuracy is required, the solver will switch to segmented ray tracing methods to obtain more accurate physically based solutions. Through this adaptive strategy, the framework seeks to balance computational efficiency and simulation fidelity in large-scale urban noise simulation.

3.3.2. Coupling with the wind solver

Following the development of the acoustic simulation framework, the noise simulator will be coupled with the urban wind simulation framework developed in the preceding chapters. Realistic wind field data generated by the wind solver will be incorporated into the acoustic simulation in order to model the influence of wind on sound propagation. In addition, the adaptive grid structures generated by the wind solver will be explored as a mechanism for determining suitable integration step sizes within the acoustic simulation. Regions with complex flow behavior or strong atmospheric gradients may require finer integration steps to preserve numerical accuracy, whereas larger step sizes may be adopted in relatively uniform regions to improve computational efficiency. Through this coupling strategy, the proposed framework aims to achieve both physically realistic sound propagation and efficient large-scale urban noise simulation.

3.4. Improving accuracy and efficiency of noise simulator using multiple importance sampling

Based on the limitation analysis of traditional importance sampling and diffuse rain methods presented in Sec. 2.2.6, this stage aims to introduce Multiple Importance Sampling (MIS) from computer graphics into urban acoustic ray tracing in order to reduce sampling variance. This subsection addresses the subRQ 4:

“To what extent can the convergence rate and accuracy of ray tracing-based urban noise simulator be improved through multiple importance sampling?”

To the best of the author’s knowledge, although MIS has been widely adopted in computer graphics rendering, it has not yet been applied to acoustic ray tracing. In this research subquestion, the author aims to introduce MIS into computational acoustics in order to mitigate the previously discussed estimation errors. Based on the analysis in Sec. 2.2.6, the author believes that MIS has the potential to provide significant improvements over existing sampling strategies. However, coupling MIS with urban noise simulation necessitates several adaptation to the traditional MIS algorithm, including coupling it with refraction models and diffraction models. The author aims to solve these problems within this research subquestion, a pipeline of this section is illustrated in Fig. 3.5.

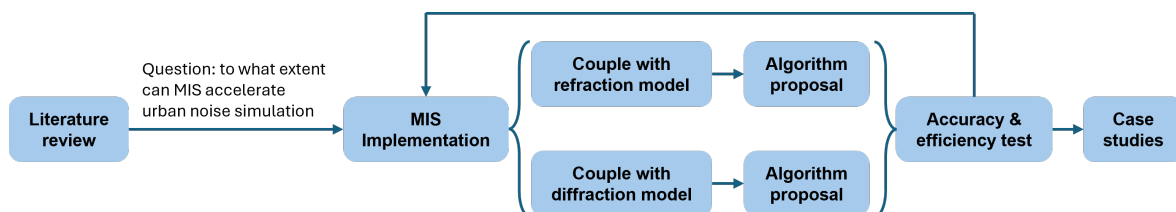


Figure 3.5: The research pipeline of sub-question 4.

3.4.1. Coupling MIS with refraction models

The first challenge concerns the coupling between MIS and refraction-aware sound ray tracing. In computer graphics and refraction-unaware sound ray tracing, rays are typically assumed to propagate along straight paths. Consequently, sound source sampling can be determined analytically. However, in refraction-aware sound ray tracing procedure, the sound source sampling procedure necessitates iterative correction and additional numerical computation due to curved propagation paths induced by atmospheric refraction. As a result, directly applying MIS may introduce significant computational overhead.

In this research subquestion, the author will focus on improving the current MIS algorithm to adapt it to refraction-aware media. Specifically, the author considers adapting the path finding technique used in eigenray tracing into sound source sampling in MIS.

3.4.2. Coupling MIS with diffraction models

Another important propagation mechanism in urban acoustics is diffraction around obstacles. Diffraction models, such as the uniform theory of diffraction (UTD) [90, 92], model the diffracted sound propagation given a ray-edge intersection. This process introduces additional ray propagation paths beyond those generated by A-BRDF sampling and source sampling. This naturally motivates a new sampling strategy based on diffraction paths. How to incorporate such a diffraction sampling strategy into the MIS framework, together with existing A-BRDF and source sampling strategies, remains an open question.

3.5. Performance and user-centered evaluation

Following the investigating of subRQs 1–4, the proposed hypotheses will be evaluated through both quantitative and qualitative analyses, as shown in Fig. 3.6. Quantitative evaluation will focus on the physical accuracy and computational performance of the proposed methods, whereas qualitative evaluation will assess their usability and effectiveness from an end-user perspective.

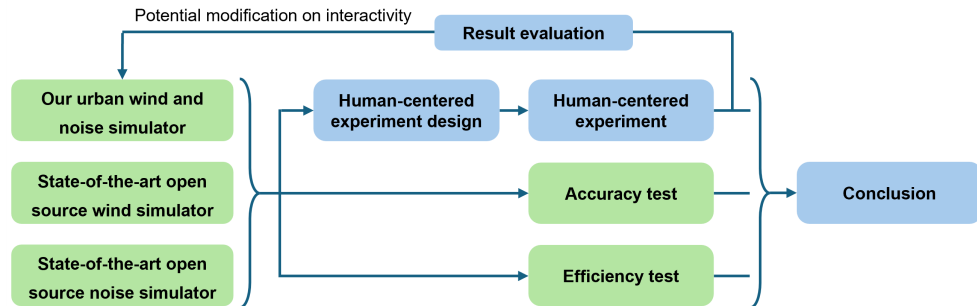


Figure 3.6: The research pipeline of sub-question 5.

3.5.1. Quantitative evaluation

The quantitative evaluation will be conducted through a series of benchmark experiments to assess the performance of the proposed framework against state-of-the-art methods.

- How does the computational efficiency of the proposed wind simulator comparing to state-of-the-art open source CFD solvers, such as OpenFOAM? This evaluation corresponds to Hypothesis 1.
- How does the proposed CFD AMR strategy perform comparing to traditional static mesh refinement methods, in terms of computational cost and solution accuracy? This evaluation corresponds to Hypothesis 4.
- How much physical fidelity can an urban noise simulator gain by coupling with an urban wind simulator, under different testing scenarios (e.g. strong / weak wind condition and simple / complex building scenario)? This evaluation corresponds to Hypothesis 5.
- How much computational efficiency improvement gain can an urban noise simulator obtain by

using compute shaders? And similarly, by using ray tracing shaders This evaluation corresponds to Hypothesis 6.

- How much computational efficiency improvement gain can an urban wind simulator obtain by using an combined strategy that uses both compute shaders and ray tracing shaders? And how much the cost is (reduced accuracy)? This evaluation corresponds to Hypothesis 6.
- Can MIS be successfully combined with refraction-aware and diffraction-aware ray tracing? If yes, how much better convergence rate gain can be obtained by integrating MIS into noise ray tracing? This evaluation corresponds to Hypothesis 7.

3.5.2. Qualitative evaluation

In addition to performance-based evaluation, the proposed framework will also be assessed from a user-centered perspective.

- How does the usability, learnability and user satisfaction of the proposed wind simulator comparing to state-of-the-art open source urban wind CFD solvers? This evaluation corresponds to Hypothesis 2 and 3.
- How does the usability and satisfaction of the proposed noise simulator comparing to state-of-the-art open source urban noise solvers? This evaluation corresponds to Hypothesis 2 and 3.

3.5.3. User-centric experiment design

To evaluate the usability and practical applicability of the proposed framework, user studies will be conducted involving participants with diverse backgrounds. Participants will be asked to perform a set of representative tasks using both the proposed software and selected baseline tools.

The study will focus primarily on workflow efficiency, interactivity, and user experience, as these aspects are difficult to assess solely through quantitative benchmarks. Example tasks may include configuring simulation scenarios and solver parameters to optimize some metrics (wind mitigation or noise reduction).

The evaluation criteria include:

- Ease of scene creation and modification
- Ease of simulation setup and solver configuration
- User confidence in the correctness of the configured simulation settings
- Clarity and interpretability of the visualization system
- Perceived efficiency of the simulation workflow
- Overall usability and user satisfaction

Where appropriate, standardized usability instruments such as the System Usability Scale (SUS) [51] and workload assessment methods such as the NASA Task Load Index (NASA-TLX) [33] may be employed to support the analysis.

Preliminary results

The first research subquestion, “How can a game engine-GIS-CFD triplet optimize the computational efficiency and usability of urban wind simulators?” has been partially addressed during the first year of this PhD project. This chapter presents the preliminary results obtained during this period.

4.1. Development of a compute shader-based large eddy simulation solver

This section presents the development of a large eddy simulation (LES) solver within the Unity game engine. First, the rationale for adopting LES and other physical models is introduced, followed by a description of its implementation in Unity. The solver is then validated and its performance is evaluated through comparisons with existing open-source CFD solvers.

4.1.1. Solver implementation

To efficiently simulating urban wind flow, achieving seamless integration of the solver and visualizer, and to facilitate wind-noise coupling in SubRQ 3 and 4, the author chose to implement LES using compute shaders within the Unity game engine. The governing equations, described in Sec. 2.1.3 are discretized using the finite volume method (FVM). A stencil-based implementation is adopted using compute shader, in which each fluid cell is processed by an individual compute shader thread mapped to a GPU thread.

Different stages of the simulation are executed sequentially using separate compute shader kernels. During execution, the CPU is primarily responsible for resource management tasks, including texture binding, kernel dispatching, and convergence monitoring, while the PDE-related computational workload is entirely offloaded to the GPU. Fig. 4.1 illustrates the overall GPU computation pipeline.

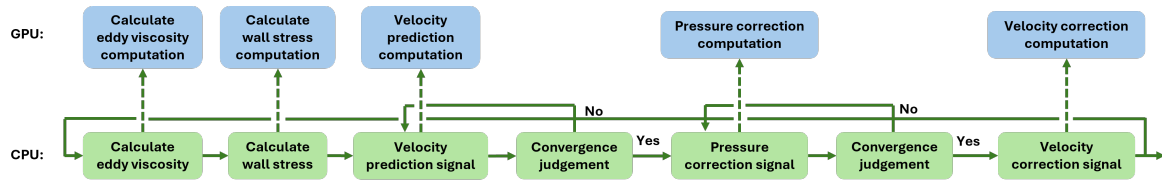


Figure 4.1: Pipeline of our stencil-based GPU fluid simulator. CPU is only responsible for sending computation signals, perform convergence judgements, and other resource management tasks.

4.1.2. Solver validation

Numerical experiments are conducted to validate the correctness of our self-implemented solver. A simplified configuration based on Test Case A from the AIJ guidelines is simulated, and the results are compared against those obtained using OpenFOAM v2312 [103]. The computational domain is shown in

Fig. 4.2. The building has dimensions $2b \times b \times b$, where b is set to $4m$. The mesh can be visualized in Fig. 4.3, with a smallest size $\Delta x = 0.2m$ around the building surface, and a stretch ratio of 1.08 towards the domain boundary. The Reynolds number calculated with the building height $2b$ and inlet velocity at $z = 10m$ is 5.333×10^6 .

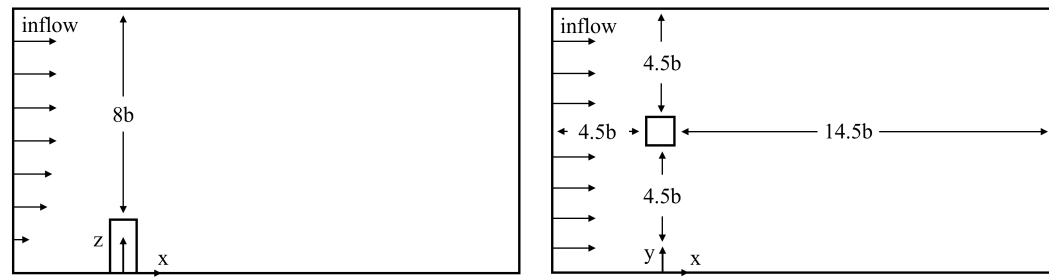


Figure 4.2: Simulation domain size (left: side view, right: top view).

The x-component velocity profile on 6 sample lines are captured with an interval of $0.05s$, and time-averaged within a $150s$ period. The approximate positions of the 6 sample lines are also visualized in Fig. 4.3. The position of Sample line (1) is $x/b = 0.75, z/b = 0.375$, sample line (2) $x/b = 2.5, z/b = 0.375$, sample line (3) $x/b = 3.25, z/b = 0.375$, sample line (4) is $x/b = 0.75, y/b = 5$, sample line (5) $x/b = 2.5, y/b = 5$, sample line (6) $x/b = 3.25, y/b = 5$.

Fig. 4.3 shows the velocity comparison between our solver and OpenFOAM. Fig. 4.3 (a) shows the time-averaged sampled velocities on line (1) and (4) for the left and right sub-plots, respectively. Fig. 4.3 (b) shows the time-averaged root-mean-square velocities on line (1) and (4). The two solvers produce consistent results in general, the small differences might result from the limited number of samples used for time averaging, and the differences in numerical treatments in the solver implementation.

4.1.3. Solver performance evaluation

Numerical experiments are conducted to evaluate the computational performance of the proposed simulator. The experiments are conducted on a workstation equipped with an Intel Core Ultra 9 185H CPU and an NVIDIA RTX 4070 Laptop GPU. Different number of processes are tested to find the most efficient setting for OpenFOAM. Experimental results in Fig. 4.4 show that the proposed solver achieves around 8.15 times speedup compared with OpenFOAM. The speedup is attributed to the fully GPU-based simulation algorithm using compute shaders in Unity game engine.

4.2. Coupling the urban wind field simulator with real world scene

After the solver was developed, validated, and benchmarked, it was integrated into large-scale urban environments within the Unity game engine. To achieve this, several components were incorporated, including scene voxelization, geographic data integration, and real-time meteorological data acquisition.

4.2.1. Scene voxelization

To perform CFD simulation within urban environments, the geometric scene needs to be converted into a voxelized representation. In this work, the scene is voxelized using a scanline-based approach, as illustrated in Fig. 4.5.

For each top layer fluid cell, a vertical ray is cast downward from the cell center using Unity's `Physics.RaycastAll` function. The intersections between the ray and the scene geometry, including buildings and terrain surfaces, are then computed with Unity's acceleration structures. Based on the intersection locations, cells above the intersection are classified as fluid cells, while cells below the surface are classified as solid cells representing buildings or terrain. This approach enables efficient voxelization of large urban scenes by leveraging Unity's built-in physics acceleration structures.

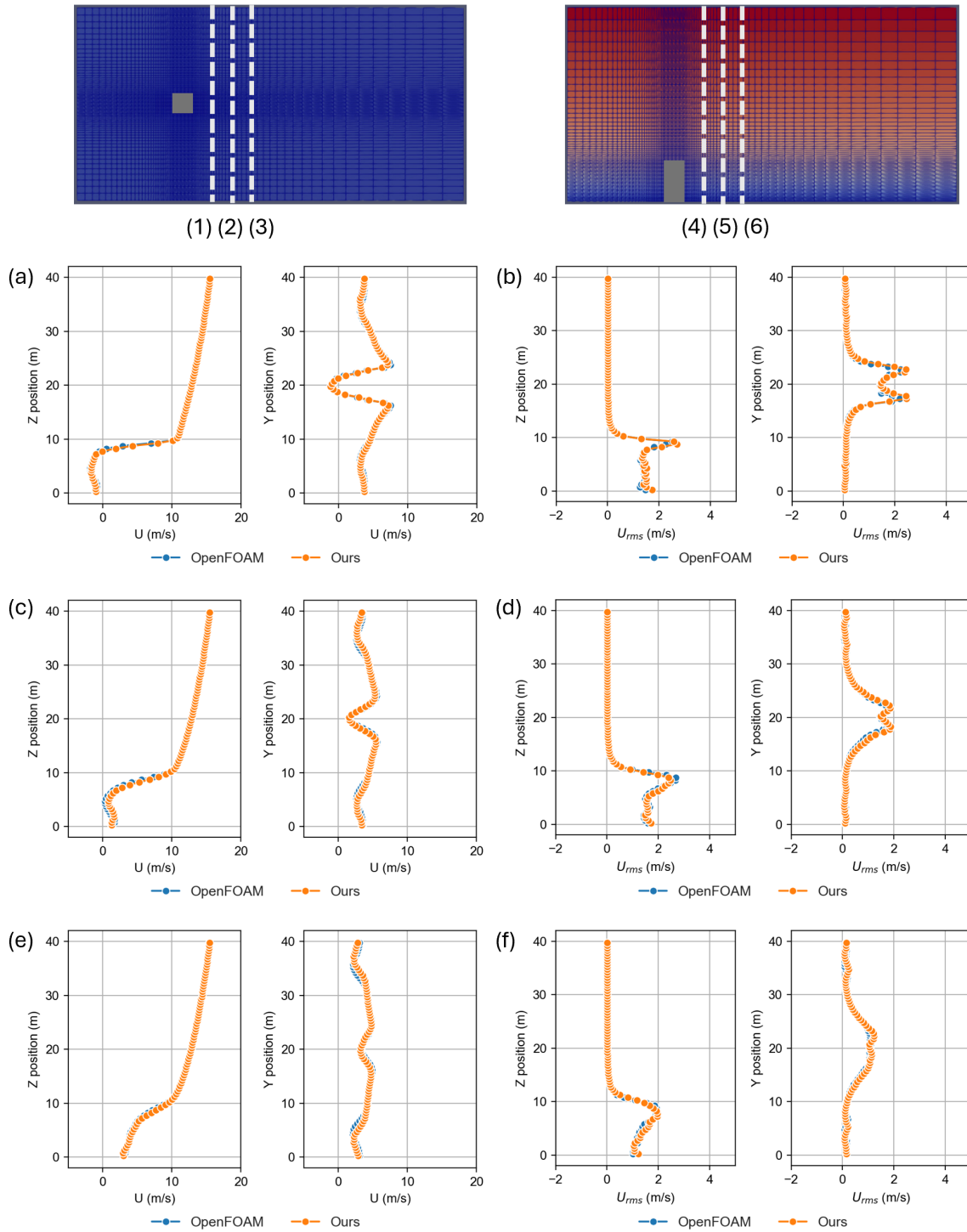


Figure 4.3: Validation of our solver. Subfigure (a) (left) and (b) (left) represents the x-component velocity and root-mean-square of x-component velocity at sample line (1), respectively. Whereas the right part of these figures represents the velocity samples at sample line (4). Subfigure (c), (d), (e), (f) follows similar rules.

4.2.2. Cesium integration

Cesium for Unity [15] is utilized to connect the wind field simulator with real-world geographic environments. It provides large-scale terrain data and OpenStreetMap-based building models within the Unity game engine [67].

During scene loading, Cesium automatically converts geographic data into Unity mesh representations. These meshes can then be directly fed into the voxelization pipeline without extra geometry processing.

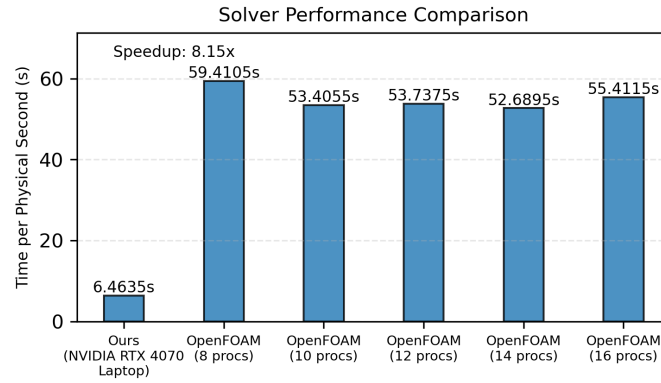


Figure 4.4: The performance of our solver compared with different settings of OpenFOAM.

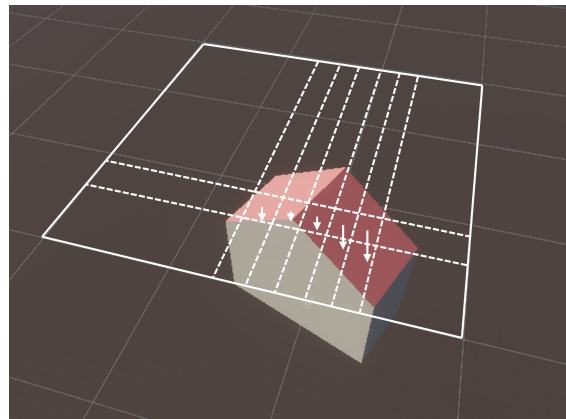


Figure 4.5: A demonstration of scanline algorithm. Rays are shot from the center of top layer fluid cells, and intersection test with the 3D scene is performed.

An example of a Cesium-integrated urban scene within Unity is shown in Fig. 4.6.

4.2.3. KNMI meteorological data integration

To couple the wind simulation with real wind speed, meteorological data provided by the Royal Netherlands Meteorological Institute (KNMI) are incorporated into the system. Specifically, near real-time wind speed and wind direction data are obtained from KNMI's *10-Minute In Situ Meteorological Observations* dataset.

During wind speed and direction query, the system obtains the wind data from the nearest four meteorological stations. The wind conditions at the target location are then estimated through interpolation of the collected station data. The interpolated wind speed and direction are subsequently used as inlet boundary condition for the CFD simulation. This design enables the simulator to dynamically adapt to changing real-world weather conditions.

4.3. Visualization

To effectively present the simulated wind field, both 2D and 3D visualization methods are implemented in our simulator.

For planar visualization, a 2D slice representation is used to display wind velocity distributions within selected sections of the computational domain. In addition, a particle-based 3D visualization based on Unity VFX is employed to illustrate the spatial structure and temporal evolution of the wind field [106].

These visualization methods provide intuitive representations of wind behavior around urban buildings. Examples of the visualization outputs are shown in Fig. 4.7.



Figure 4.6: A demonstration of Cesium for Unity scene.

4.4. Asynchronous data exportation

In order to export the simulation results without interrupting the simulation process, an asynchronous data exportation pipeline based on the NetCDF4 file format is developed.

The core of the pipeline is a first-in-first-out (FIFO) queue used for inter-thread communication. When the main simulation thread triggers a data storage operation, the simulation data are first copied from GPU textures to CPU memory in an asynchronous manner, and subsequently pushed into the FIFO queue. A dedicated data-saving thread is responsible for monitoring the queue and writes the buffered data into NetCDF files.

With this design, the computationally expensive file writing operations are fully offloaded to a secondary CPU thread, thereby minimizing interruptions to the simulation pipeline and improving runtime performance. An overview of the asynchronous exportation pipeline is shown in Fig. 4.8.

4.5. Web visualizer

To facilitate online visualization and sharing of simulation results, a web-based visualization system is developed using the trame framework [43], as illustrated in Fig. 4.9.

The current web visualizer can store simulated wind field and related building geometries on the server. It enables users to interactively explore the simulated wind field through a standard web browser, without requiring installation of specialized solver. This design improves the accessibility and usability of the simulation platform for both researchers and non-expert users.

The web visualizer is publicly accessible at: <https://xuanchenzhou-windfieldvisualizer.hf.space/>.

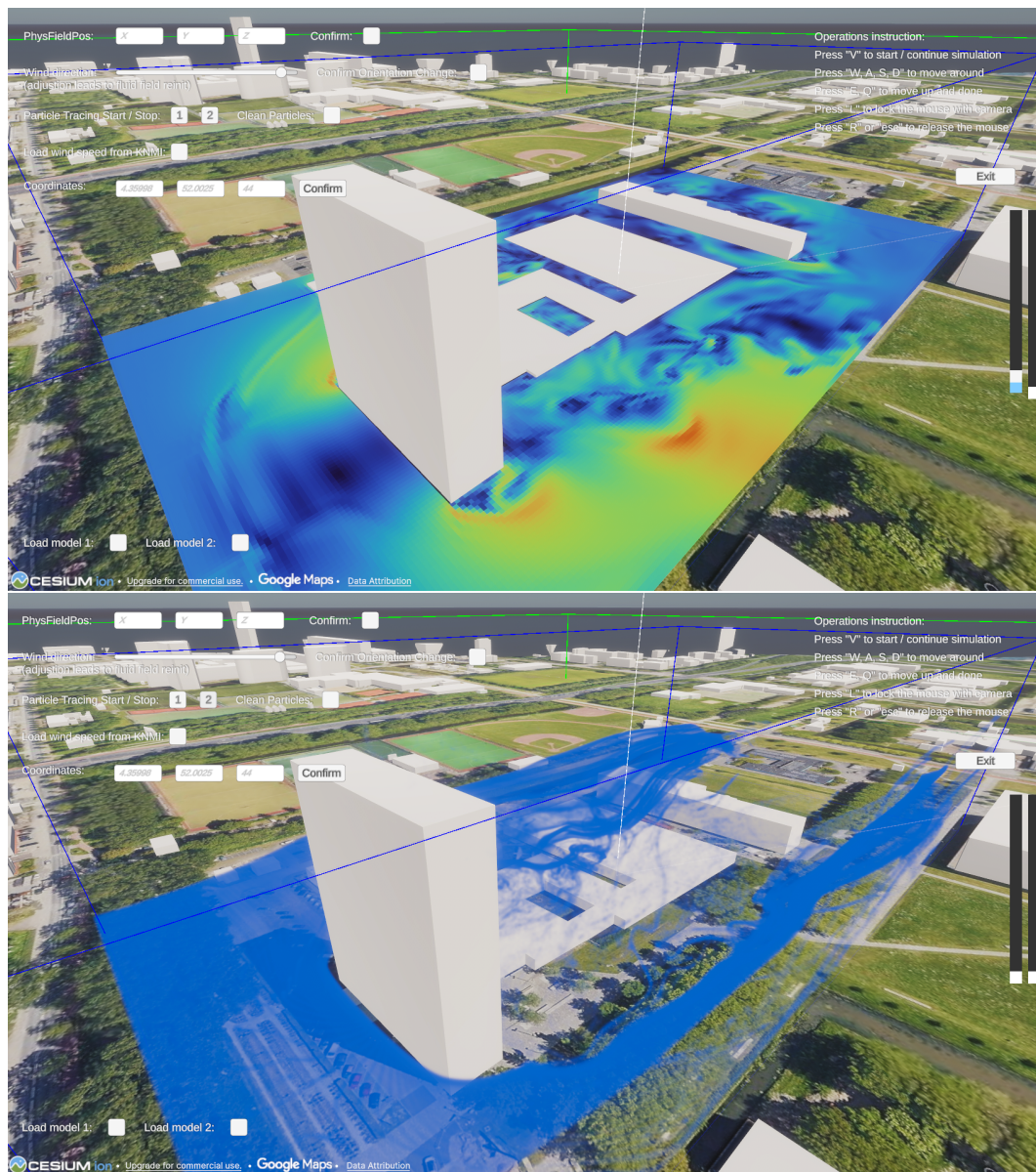


Figure 4.7: (Top) 2D velocity field visualization, (bottom) 3D particle-based fluid field visualization.

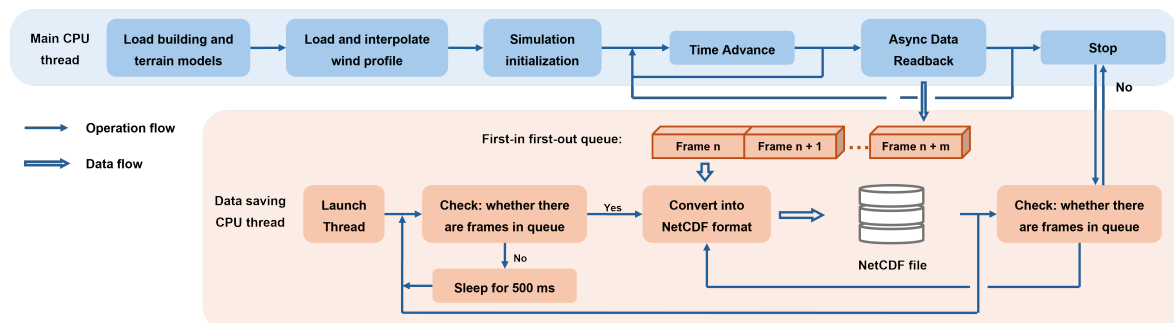


Figure 4.8: Asynchronous data saving pipeline.

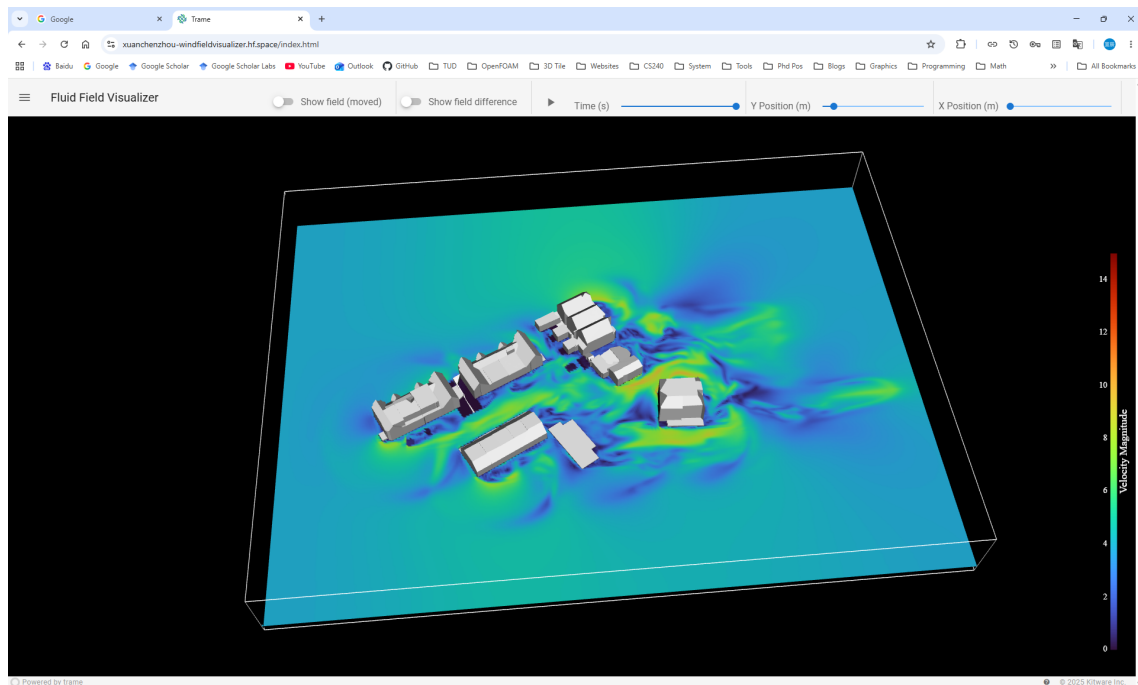


Figure 4.9: The web-based visualizer.

5

Plan

5.1. Supervision and project team

This PhD project will be carried out by the author, who is responsible for the detailed planning, implementation, evaluation and paper writing of the research. The author is supervised by Prof.dr.ir. Peter van Oosterom (Promotor) and Dr. Azaraskh Rafiee (Daily supervisor). The supervisory team provides research direction guidance, scientific guidance, technical support and paper revision throughout the research process.

The progress meeting between the author and the supervisors are scheduled on a bi-weekly basis. These meetings provide an opportunity to review the completed work, discuss ongoing research activities, identify challenges, and revise future plans of the project. The meetings also serve to monitor the project timeline.

In addition to the bi-weekly progress meetings, the author will also participate in monthly GIS group meetings, and potential seminars. Feedbacks obtained from these activities will contribute to the continuous improvement of the research.

Moreover, this research may benefit from interactions with the POINT-TWINS project [19]. POINT-TWINS investigates mesh-free numerical methods for urban wind simulation, and develop new methods of massive 3D point cloud data management. Given the closely related research directions, opportunities for knowledge exchange and collaboration may emerge in areas such as geospatial data management and adaptive mesh refinement for urban wind simulation.

5.2. Resources and tools

This PhD project will utilize open datasets as input. Related data sources include:

- OpenStreetMap buildings data [67]
- 3DBAG buildings data [73]
- Cesium terrain data [15]
- AHN [1]
- KNMI [86]
- Nationaal Dataportaal Wegverkeer (NDW) [61]
- Nationaal Wegen Bestand (NWB) [83]
- Prorail [76]
- Luchtverkeersleiding Nederland (LVNL) [54]

Needed software include:

- Unity

Needed programming languages include:

- C#
- HLSL
- Python

5.3. Publication plan

According to the PhD agreement, 3 journal papers are planned corresponding to the research targets. The first journal paper focus on the integration of GIS and CFD into game engine, building an interactable and efficient urban wind simulator. The publication target of the first paper is journal about environmental modelling software. The current planned journal to be submitted is:

- Environmental Modelling & Software

The second journal paper focus on discussing to what extent can adaptive mesh refinement increase the efficiency of urban wind simulation. The publication target of the second paper is journal about building wind model or applied computational fluid dynamics. Those three journals are possible targets:

- Computer Physics Communications
- Building and Environment
- Computers, Environment and Urban Systems
- Computers & Geosciences
- Journal of Wind Engineering and Industrial Aerodynamics

The third journal paper focus on GPU / shader-accelerated urban noise simulation and its combination with urban wind simulator. Another focus on the third journal paper (or a separate paper) is about improving the efficiency of noise simulator using multiple importance sampling. The publication target is journal about computational acoustics, high performance computing, and their applications in building and urban region. Related publication targets include:

- Building and Environment
- Computers, Environment and Urban Systems
- Computers & Geosciences
- The Journal of the Acoustical Society of America
- Applied Acoustics
- Acta Acustica
- Computer Physics Communications

The fourth journal paper (or integrated with the former three papers) focus on validation and human-centered evaluation of the proposed software. The publication target of this paper include:

- Environment and Planning B: Urban Analytics and City Science
- Environmental Modelling & Software
- Computers, Environment and Urban Systems

5.4. Conference plan

According to the PhD agreement, 6 conference papers are planned during the PhD. The conference attendance plan is: 1 conference attendance for the 1st year, 2 for the 2nd year, 2 for the 3rd year and 1 for the 4th year.

First year:

Accepted conference:

- AGILE conference, Tartu, Estonia, June 2026 (best paper candidate).

Second year:

Possible conference:

- International Congress on Environmental Modelling and Software (IEMSS) (software developing oriented).
- The European Community on Computational Methods in Applied Sciences (ECCOMAS) (computational physics oriented).
- European Fluid Dynamics Conference (EFDC3) (computational physics oriented).
- Platform for Advanced Scientific Computing (PASC) (parallel computation oriented).
- Forum Acusticum (computational acoustics oriented).

Third year:

Possible conference:

- Symposium on Simulation for Architecture and Urban Design (SimAUD) (simulation & urban design oriented).
- International Congress on Acoustics (computational acoustics oriented).
- Forum Acusticum (computational acoustics oriented).
- Eurographics (ray tracing algorithm oriented).
- High-Performance Graphics (ray tracing algorithm oriented).

Fourth year:

Possible conference:

- International Congress on Environmental Modelling and Software (IEMSS) (software developing oriented).
- Forum Acusticum (computational acoustics oriented).
- Eurographics (ray tracing algorithm oriented).
- High-Performance Graphics (ray tracing algorithm oriented).

5.5. Master's thesis supervision plan

The author is currently supervising a Master student. The author is expected to supervise Master's thesis within in these related topics:

- Computational fluid dynamics-based urban wind simulation.
- Computational acoustics-based urban sound simulation.
- Physical simulation and rendering in game engine.

5.6. Graduate school course planning

The accomplished and planned courses during the PhD project are listed in Table 5.1.

* Lotj: Learning on the Job

5.7. Project timeline

The planned project timeline is listed in Table 5.2.

5.8. Self-reflection of the first year

My first year as a PhD student has been both busy but rewarding. Looking back, I believe I have made substantial progress in both my research project and my personal development.

Regarding the research project, I primarily focused on the first research subquestion, which investigates the integration of game engines, GIS, and CFD for urban wind simulation. I think I have made substantial progress on this topic. The resulting research outcome has been accepted as a conference paper (best paper candidate), and an extended journal paper is currently under preparation.

In addition to the completed work, I have also developed plans and conducted literature reviews for the remaining research questions. Through this process, I have gained a clear understanding of the research context of my PhD project. I believe the proposed research directions are feasible and offer promising scientific values. I am very excited about them.

For personal development, I have improved both my technical knowledge and my communication skills.

For knowledge improvement, I completed a GIS-related course during this year, which strengthened my understanding of geographic information systems and spatial data processing. I also audited a course on computational fluid dynamics, which improved my understanding in advanced turbulence modeling techniques in CFD.

I have also had several opportunities to present my work to both our research group and the department. Through these presentations and the feedback I received, I have become more confident in communicating research ideas and have developed a better understanding of how to deliver clear and effective scientific presentations.

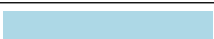
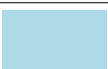
One aspect that I would like to improve in the coming years is the efficiency of my academic writing and revision process. During the past year, I found that preparing the conference paper, responding to reviewers' comments, and writing this Go / No-Go report required more time than I had initially anticipated. I believe there is still room to streamline my writing workflow and revision process. Improving in this area would allow me to devote more time to experiments, algorithm development, and other research activities.

Overall, I am satisfied with the progress I have made during my first year as a PhD student. The experiences of the past year have provided me with a solid foundation for the remainder of my PhD, and I look forward to further advancing this research in the coming years.

Table 5.1: Graduate school course planning

	Completed	In progress	Planned
Discipline-related skills			
GEO1002 Geographical Information Systems and Cartography	5		
ABE009 Pitching Your PhD: Getting ready for the Go/No-Go	4		
Attending a seminar		1-2	
AE4431-23 Aircraft noise			3
ABE022 Academic Writing Retreat			4
Research skills			
R2.B3 Data Visualization as a Tool for Scientific Research in Python	1		
R4.A1 Research Data Management 101	2		
LotJ Writing the first conference paper	1		
LotJ Supervising a MSc student / BSc project groups		2-6	
LotJ Writing an international, peer-reviewed journal article			2-4
LotJ Addressing a major international audience			1
R2.B1 Speedreading and Mindmapping			1.5
R2.B2 Statistics for PhD Research			4
Transferable skills			
T1.A7 Data Visualization - A Practical Approach	1		
T4.G1 PhD startup course A1	0.5		
T4.G1 PhD startup course A2	0.5		
T4.G1 PhD startup course A3	0.5		
T4.G1 PhD startup course B	0.5		
T1.B1 Creative tools for scientific writing			3.5
T3.B1 Coaching Individual Students			1
T1.D2 Academic English 2			4
T1.D9 Dutch for non native speakers (NT2)			4

Table 5.2: Research timeline

SubRQs	Task	Year 1	Year 2	Year 3	Year 4
1	Literature review				
	GPU accelerated LES urban wind simulator implementation				
	Game engine-GIS-CFD integration				
2	Literature review				
	AMR acceleration of the urban wind simulator, and explore the suitable criteria of different algorithms				
3	Literature review				
	Stochastic, wind coupled ray tracing urban noise simulator implementation on GPU				
	Game engine-GIS-noise simulation integration				
4	Literature review				
	Explore adapting MIS on urban noise simulator				
	Explore incorporating MIS with refraction and diffraction models				
5	Quantitative evaluation of the overall system against state-of-the-art methods				
	User-centered evaluation of usability, interactivity and satisfaction				
	Finalizing the thesis				
	Defense				

References

- [1] Actueel Hoogtebestand Nederland. *AHN5 Dataset*. <https://www.ahn.nl>. 2026.
- [2] Advanced Micro Devices, Inc. *ROCm Open Software Platform*. Version 6.1. 2024. URL: <https://github.com/ROCm/ROCm>.
- [3] Pouria Amirian, Anahid Basiri, and Adam Winstanley. "Evaluation of data management systems for geospatial big data". In: *International Conference on Computational Science and Its Applications*. Springer. 2014, pp. 678–690.
- [4] Dionysios Angelidis, Vasiliki D Assimakopoulos, and George Bergeles. "A Cartesian grid refinement method for simulating thermally stratified urban environments". In: *Journal of Wind Engineering and Industrial Aerodynamics* 142 (2015), pp. 149–163.
- [5] Christophe Bailly and Daniel Juve. "Numerical solution of acoustic propagation problems using linearized Euler equations". In: *AIAA journal* 38.1 (2000), pp. 22–29.
- [6] Marsha J Berger and Phillip Colella. "Local adaptive mesh refinement for shock hydrodynamics". In: *Journal of computational Physics* 82.1 (1989), pp. 64–84.
- [7] Marsha J Berger and Joseph Oliger. "Adaptive mesh refinement for hyperbolic partial differential equations". In: *Journal of computational Physics* 53.3 (1984), pp. 484–512.
- [8] Matthias Berger and Verina Cristie. "CFD post-processing in Unity3D". In: *Procedia Computer Science* 51 (2015), pp. 2913–2922.
- [9] Birgitta Berglund et al. "Guidelines for community noise". In: (1999).
- [10] Haoyu Bian et al. "An efficient rectilinear Gaussian beam tracing method for sound propagation modelling in a non-turbulent medium". In: *The Journal of the Acoustical Society of America* 148.6 (2020), pp. 4037–4048.
- [11] Wojciech Binek, Adam Pilch, and Tadeusz Kaminski. "Direct application of the diffusers' reflection patterns in geometrical acoustics simulations". In: *Applied Acoustics* 198 (2022), p. 108949.
- [12] Bert Blocken, WD Janssen, and Twan van Hooff. "CFD simulation for pedestrian wind comfort and wind safety in urban areas: General decision framework and case study for the Eindhoven University campus". In: *Environmental Modelling & Software* 30 (2012), pp. 15–34.
- [13] Alan Lex Brown and Irene Van Kamp. "WHO environmental noise guidelines for the European region: A systematic review of transport noise interventions and their impacts on health". In: *International journal of environmental research and public health* 14.8 (2017), p. 873.
- [14] Vlastislav Cerveny. *Seismic ray theory*. 2001.
- [15] Cesium GS, Inc. *Cesium for Unity*. <https://github.com/CesiumGS/cesium-unity>. Version v1.19.0. Accessed: 2026-02-18. 2025.
- [16] Kang-Tsung Chang. *Introduction to geographic information systems*. Vol. 4. McGraw-Hill Boston, 2008.
- [17] Ruoping Chu and Kai Wang. "CFD in Urban Wind Resource Assessments: A Review". In: *Energies* 18.10 (2025), p. 2626.
- [18] Tim Colonius and Sanjiva K Lele. "Computational aeroacoustics: progress on nonlinear problems of sound generation". In: *Progress in Aerospace sciences* 40.6 (2004), pp. 345–416.
- [19] Web Editorial Communication. *POINT-TWINS: Point cLOUD-based dynamIc cONTinuous-scale adapTive urban WIND Simulation*. Accessed: 2026-06-04. 2026. URL: <https://www.tudelft.nl/en/2025/tu-delft/open-technology-programme-funds-again-two-delft-research-projects>.

- [20] Brent Cowan and Bill Kapralos. "GPU-based real-time acoustical occlusion modeling". In: *Virtual reality* 14.3 (2010), pp. 183–196.
- [21] James W Deardorff. "A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers". In: *Journal of Fluid Mechanics* 41.2 (1970), pp. 453–480.
- [22] Martina E Deininger et al. "A continuous, semi-automated workflow: From 3D city models with geometric optimization and CFD simulations to visualization of wind in an urban environment". In: *ISPRS International Journal of Geo-Information* 9.11 (2020), p. 657.
- [23] Alexey Dosovitskiy et al. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.
- [24] Guillaume Dutilleul et al. "NMPB-routes-2008: The revision of the french method for road traffic noise prediction". In: *Acta Acustica united with Acustica* 96.3 (2010), pp. 452–462.
- [25] Epic Games. *Niagara Visual Effects System*. Version 5.7. Epic Games, 2026. URL: <https://dev.epicgames.com/documentation/unreal-engine/creating-visual-effects-in-niagara-for-unreal-engine>.
- [26] Ugo Finnendahl et al. "Differentiable geometric acoustic path tracing using time-resolved path replay backpropagation". In: *ACM Transactions on Graphics (TOG)* 44.4 (2025), pp. 1–17.
- [27] Jörg Franke et al. "Best practice guideline for the CFD simulation of flows in the urban environment". PhD thesis. COST European Cooperation in Science and Technology, 2007.
- [28] Gemeente Delft. *Open Data Portaal Delft*. Accessed: 2026-05-31. 2026. URL: <https://data.delft.nl/>.
- [29] Giorgio Giangaspero et al. "High-order scale-resolving simulations of extreme wind loads on a model high-rise building". In: *Journal of Wind Engineering and Industrial Aerodynamics* 230 (2022), p. 105169.
- [30] Akshay A Gowardhan et al. "A CFD-based wind solver for an urban fast response transport and dispersion model". In: *Environmental fluid mechanics* 11.5 (2011), pp. 439–464.
- [31] Bomenstichting Den Haag. *Open Data Bomen in Den Haag*. Accessed: 2026-05-31. 2026. URL: <https://www.bomenstichtingdenhaag.nl/groen-in-de-gemeente-den-haag/open-data/>.
- [32] Trond-Ola Hågbo and Knut Erik Teigen Giljarhus. "Sensitivity of urban morphology and the number of CFD simulated wind directions on pedestrian wind comfort and safety assessments". In: *Building and Environment* 253 (2024), p. 111310.
- [33] Sandra G Hart. "NASA-task load index (NASA-TLX); 20 years later". In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 50. 9. Sage publications Sage CA: Los Angeles, CA. 2006, pp. 904–908.
- [34] Maarten Hornikx, Thomas Krijnen, and Louis van Harten. "openPSTD: The open source pseudo-spectral time-domain method for acoustic propagation". In: *Computer Physics Communications* 203 (2016), pp. 298–308.
- [35] Qing Hou, Ming Cai, and Haibo Wang. "Dynamic modeling of traffic noise in both indoor and outdoor environments by using a ray tracing method". In: *Building and Environment* 121 (2017), pp. 225–237.
- [36] Yongyu Hu, Fusuo Xu, and Zhi Gao. "A comparative study of the simulation accuracy and efficiency for the urban wind environment based on CFD plug-ins integrated into architectural design platforms". In: *Buildings* 12.9 (2022), p. 1487.
- [37] M Imano et al. "Adaptive Mesh Refinement Method Suitable for CFD Analysis of Wind Environment around Buildings". In: (2008).
- [38] Khodr Jaber, Ebenezer E Essel, and Pierre E Sullivan. "GPU-native adaptive mesh refinement with application to lattice Boltzmann simulations". In: *Computer Physics Communications* 311 (2025), p. 109543.

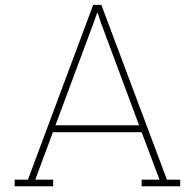
- [39] Dorota Jarosińska et al. "Development of the WHO environmental noise guidelines for the European region: an introduction". In: *International journal of environmental research and public health* 15.4 (2018), p. 813.
- [40] Zitao Jiang et al. "A literature review of cross ventilation in buildings". In: *Energy and Buildings* 291 (2023), p. 113143.
- [41] Yi Jing et al. "Quantitative city ventilation evaluation for urban canopy under heat island circulation without geostrophic winds: Multi-scale CFD model and parametric investigations". In: *Building and environment* 196 (2021), p. 107793.
- [42] Richard Michael Jones. *HARPA: A versatile three-dimensional Hamiltonian ray-tracing program for acoustic waves in the atmosphere above irregular terrain*. US Department of Commerce, National Oceanic and Atmospheric Administration . . ., 1986.
- [43] S. Jourdain, P. O'Leary, and W. Schroeder. "Trame: Platform Ubiquitous, Scalable Integration Framework for Visual Analytics". In: *IEEE Computer Graphics and Applications* (Mar. 2025). doi: 10.1109/MCG.2025.3540264.
- [44] Yu-Hsuan Juan et al. "CFD assessment of wind energy potential for generic high-rise buildings in close proximity: Impact of building arrangement and height". In: *Applied Energy* 321 (2022), p. 119328.
- [45] Zeynab Kaseb and Morteza Rahbar. "Towards CFD-based optimization of urban wind conditions: Comparison of Genetic algorithm, Particle Swarm Optimization, and a hybrid algorithm". In: *Sustainable Cities and Society* 77 (2022), p. 103565.
- [46] Zeynab Kaseb et al. "A framework for pedestrian-level wind conditions improvement in urban areas: CFD simulation and optimization". In: *Building and Environment* 184 (2020), p. 107191.
- [47] Patrick Kastner and Timur Dogan. "Eddy3D: A toolkit for decoupled outdoor thermal comfort simulations in urban areas". In: *Building and Environment* 212 (2022), p. 108639.
- [48] Stylianos Kephalopoulos et al. "Advances in the development of common noise assessment methods in Europe: The CNOSSOS-EU framework for strategic environmental noise mapping". In: *Science of the Total Environment* 482 (2014), pp. 400–410.
- [49] Eric Lafortune. "Mathematical models and Monte Carlo algorithms for physically based rendering". In: *Department of Computer Science, Faculty of Engineering, Katholieke Universiteit Leuven* 20.74-79 (1996), p. 4.
- [50] M Lateb et al. "On the use of numerical modelling for near-field pollutant dispersion in urban environments- A review". In: *Environmental Pollution* 208 (2016), pp. 271–283.
- [51] James R Lewis. "The system usability scale: past, present, and future". In: *International Journal of Human-Computer Interaction* 34.7 (2018), pp. 577–590.
- [52] Xiaodong Li, Steven Hulshoff, and Stefan Hickel. "Mesh adaptation using adjoint methods and reduced-order models for large eddy simulation". In: *14th WCCM-ECCOMAS Congress, CIMNE*. 2021.
- [53] Yezhan Li and Tsubasa Okaze. "Application of polyhedral meshes in large-eddy simulation of building array flow fields within neutral and unstable boundary layers". In: *Building and Environment* 263 (2024), p. 111846.
- [54] Luchtverkeersleiding Nederland. *Luchtverkeersleiding Nederland*. Accessed: 2026-05-15. 2026. URL: <https://www.lvn1.nl>.
- [55] Zhihan Lv et al. "Game on, science-how video game technology may help biologists tackle visualization challenges". In: *PloS one* 8.3 (2013), e57990.
- [56] Sean McTavish, Hali Barber, and Alanna Wall. "Validation of urban airflow measurements through a combined field test and wind tunnel study". In: *Journal of Wind Engineering and Industrial Aerodynamics* 265 (2025), p. 106155.

- [57] Enzo Meneses et al. "Ray Tracing Cores for General-Purpose Computing: A Literature Review". In: *arXiv preprint arXiv:2603.28771* (2026).
- [58] Qi Mo et al. "Analytic ray curve tracing for outdoor sound propagation". In: *Applied Acoustics* 104 (2016), pp. 142–151.
- [59] Qi Mo et al. "Outdoor sound propagation with analytic ray curve tracer and Gaussian beam". In: *The Journal of the Acoustical Society of America* 141.3 (2017), pp. 2289–2299.
- [60] Fabio Murena et al. "Modelling dispersion of traffic pollution in a deep street canyon: Application of CFD and operational models". In: *Atmospheric Environment* 43.14 (2009), pp. 2303–2311.
- [61] Nationaal Dataportaal Wegverkeer (NDW). *Real-time data*. Dataset. Accessed: 2026-05-15. 2026. URL: <https://opendata.ndw.nu>.
- [62] Tsuyoshi Nozu et al. "Mesh-adaptive LES for wind load estimation of a high-rise building in a city". In: *Journal of Wind Engineering and Industrial Aerodynamics* 144 (2015), pp. 62–69.
- [63] Nvidia. *Nvidia PhysX*. Accessed: 2026-05-19. 2026. URL: https://nvidia-omniverse.github.io/PhysX/physx/5.8.0/_api_build/classPxBVH.html#_CPPv4N5PxBVHD0Ev.
- [64] NVIDIA Corporation. *NVIDIA CUDA Toolkit Documentation*. Version 12.4. 2024. URL: <https://docs.nvidia.com/cuda/>.
- [65] Tsubasa Okaze et al. "AIJ guidelines on the applications of large-eddy simulation to pedestrian wind environment: Recommendations and validation benchmarks". In: *Journal of Wind Engineering and Industrial Aerodynamics* 269 (2026), p. 106321.
- [66] OpenCFD Ltd. *OpenFOAM Version v2312*. ESI Group. Bracknell, UK, 2023. URL: <https://www.openfoam.com/>.
- [67] OpenStreetMap contributors. *Planet dump retrieved from https://planet.osm.org*. <https://www.openstreetmap.org>. 2026.
- [68] Vladimir E Ostashev et al. "Equations for finite-difference, time-domain simulation of sound propagation in moving inhomogeneous media and numerical implementation". In: *The Journal of the Acoustical Society of America* 117.2 (2005), pp. 503–517.
- [69] Emile OUELLET-DELOREME et al. "Live ray tracing and auralization of 3d audio scenes with". In: (2021).
- [70] Mariya Pantusheva et al. "Towards a Comprehensive Workflow for Mesh Generation in Urban Wind Engineering using CFD". In: *Journal of Physics: Conference Series*. Vol. 3027. 1. IOP Publishing. 2025, p. 012079.
- [71] Stephanie M Pasareanu, Ricardo A Burdisso, and Marcel C Remillieux. "A numerical hybrid model for outdoor sound propagation in complex urban environments". In: *The Journal of the Acoustical Society of America* 143.3 (2018), EL218–EL224.
- [72] Suhas V Patankar and D Brian Spalding. "A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows". In: *Numerical prediction of flow, heat transfer, turbulence and combustion*. Elsevier, 1983, pp. 54–73.
- [73] Ravi Peters et al. "Automated 3D reconstruction of LoD2 and LoD1 models for all 10 million buildings of the Netherlands". In: *Photogrammetric Engineering and Remote Sensing* 88.3 (2022), pp. 165–170. DOI: 10.14358/PERS.21-00032R2.
- [74] Ugo Piomelli. "Large-eddy simulation: achievements and challenges". In: *Progress in aerospace sciences* 35.4 (1999), pp. 335–362.
- [75] Alexander Pohl. "Simulation of diffraction based on the uncertainty relation". In: (2014).
- [76] ProRail. *ProRail*. Accessed: 2026-05-15. 2026. URL: <https://www.prorail.nl/samenwerken/leveranciers/praktische-informatie/spoordata>.

- [77] John M Prospathopoulos and Spyros G Voutsinas. "Application of a ray theory model to the prediction of noise emissions from isolated wind turbines and wind parks". In: *Wind Energy: An International Journal for Progress and Applications in Wind Power Conversion Technology* 10.2 (2007), pp. 103–119.
- [78] Xinsheng Qin, Randall J LeVeque, and Michael R Motley. "Accelerating an adaptive mesh refinement code for depth-averaged flows using GPUs". In: *Journal of Advances in Modeling Earth Systems* 11.8 (2019), pp. 2606–2628.
- [79] Azarakhsh Rafiee et al. "Developing a wind turbine planning platform: Integration of "sound propagation model-GIS-game engine" triplet". In: *Environmental Modelling & Software* 95 (2017), pp. 326–343.
- [80] CF Ratto et al. "Mass-consistent models for wind fields over complex terrain: the state of the art". In: *Environmental Software* 9.4 (1994), pp. 247–268.
- [81] Marcel C Remillieux et al. "Experimental and numerical study on the propagation of impulsive sound around buildings". In: *Applied Acoustics* 73.10 (2012), pp. 1029–1044.
- [82] Osborne Reynolds. "IV. On the dynamical theory of incompressible viscous fluids and the determination of the criterion". In: *Philosophical transactions of the royal society of london.(a.)* 186 (1895), pp. 123–164.
- [83] Rijkswaterstaat. *Nationaal Wegenbestand (NWB) - Wegen - Wegvakken en Hectometerpunten*. Dataset, PDOK. Accessed: 2026-05-15. 2026. URL: <https://www.pdok.nl/introductie/-/article/nationaal-wegen-bestand-nwb-wegen>.
- [84] Riccardo Rossi et al. "Parallel adaptive mesh refinement for incompressible flow problems". In: *Computers & Fluids* 80 (2013), pp. 342–355.
- [85] Mostapha Sadeghipour Roudsari and Michelle Pak. "Ladybug: a parametric environmental plugin for grasshopper to help designers create an environmentally-conscious design". In: (2013).
- [86] Royal Netherlands Meteorological Institute (KNMI). *[Insert Specific Dataset Name, e.g., Daily observational data from KNMI weather stations]*. <https://dataplatform.knmi.nl>. 2026.
- [87] Erik Salomons et al. "The Harmonoise sound propagation model". In: *Acta acustica united with acustica* 97.1 (2011), pp. 62–74.
- [88] Erik M Salomons. *Computational atmospheric acoustics*. Springer Science & Business Media, 2001.
- [89] Beatriz Sanchez et al. "Modelling NOX concentrations through CFD-RANS in an urban hot-spot using high resolution traffic emissions and meteorology from a mesoscale model". In: *Atmospheric Environment* 163 (2017), pp. 155–165.
- [90] Lauri Savioja and U Peter Svensson. "Overview of geometrical room acoustic modeling techniques". In: *The Journal of the Acoustical Society of America* 138.2 (2015), pp. 708–730.
- [91] Philipp Schäfer and Michael Vorländer. "Atmospheric Ray Tracing: An efficient, open-source framework for finding eigenrays in a stratified, moving medium". In: *Acta Acustica* 5 (2021), p. 26.
- [92] Carl Schissler, Gregor Mückl, and Paul Calamia. "Fast diffraction pathfinding for dynamic sound propagation". In: *ACM Transactions on Graphics (TOG)* 40.4 (2021), pp. 1–13.
- [93] Dirk Schröder. *Physically based real-time auralization of interactive virtual environments*. Vol. 11. Logos Verlag Berlin GmbH, 2011.
- [94] Dirk Schröder and Alexander Pohl. "Real-time hybrid simulation method including edge diffraction". In: *EAA auralization symposium*. Vol. 39. 2009.
- [95] Usman Shaukat et al. "Validation of large eddy simulations in urban wind studies using a new overall area metric". In: *Building and Environment* 267 (2025), p. 112285.
- [96] Weijing Shi et al. "Algorithm and hardware implementation for visual perception system in autonomous vehicle: A survey". In: *Integration* 59 (2017), pp. 148–156.

- [97] Mohammadreza Shirzadi and Yoshihide Tominaga. "CFD evaluation of mean and turbulent wind characteristics around a high-rise building affected by its surroundings". In: *Building and Environment* 225 (2022), p. 109637.
- [98] Balwinder Singh et al. "Evaluation of the QUIC-URB fast response urban wind model for a cubical building array and wide building street canyon". In: *Environmental fluid mechanics* 8.4 (2008), pp. 281–312.
- [99] Mohammad Taleghani et al. "Thermal assessment of heat mitigation strategies: The case of Portland State University, Oregon, USA". In: *Building and Environment* 73 (2014), pp. 138–150.
- [100] Jia-Hong Tang et al. "Dynamic modeling for noise mapping in urban areas". In: *Environmental Impact Assessment Review* 97 (2022), p. 106864.
- [101] Lonny L Thompson. "A review of finite-element methods for time-harmonic acoustics". In: *The Journal of the Acoustical Society of America* 119.3 (2006), pp. 1315–1330.
- [102] Francisco Toja-Silva et al. "A review of computational fluid dynamics (CFD) simulations of the wind flow around buildings for urban wind energy exploitation". In: *Journal of Wind Engineering and Industrial Aerodynamics* 180 (2018), pp. 66–87.
- [103] Yoshihide Tominaga et al. "AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings". In: *Journal of wind engineering and industrial aerodynamics* 96.10-11 (2008), pp. 1749–1761.
- [104] Paul G Tucker. *Unsteady computational fluid dynamics in aeronautics*. Vol. 104. Springer Science & Business Media, 2013.
- [105] Unity. *Unity uGUI*. Accessed: 2026-05-19. 2026. URL: <https://docs.unity3d.com/Packages/com.unity.ugui@2.6/manual/index.html>.
- [106] Unity Technologies. *Visual Effect Graph*. Version 17.6.0. Unity Technologies, 2026. URL: <https://docs.unity3d.com/Packages/com.unity.visualeffectgraph@17.6/manual/index.html>.
- [107] Rafael Valencia-García et al. "Technologies and innovation: Second international conference, citi 2016, guayaquil, ecuador, november 23-25, 2016". In: *Proceedings. Communications in Computer and Information Science* 658 (2016).
- [108] Elise Van Kempen et al. "WHO environmental noise guidelines for the European region: a systematic review on environmental noise and cardiovascular and metabolic effects: a summary". In: *International journal of environmental research and public health* 15.2 (2018), p. 379.
- [109] Eric Veach and Leonidas J Guibas. "Optimally combining sampling techniques for Monte Carlo rendering". In: *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. 1995, pp. 419–428.
- [110] Guglielmo Vivarelli, Ning Qin, and Shahrokh Shahpar. "A review of mesh adaptation technology applied to computational fluid dynamics". In: *Fluids* 10.5 (2025), p. 129.
- [111] Haibo Wang, Huimin Gao, and Ming Cai. "Simulation of traffic noise both indoors and outdoors based on an integrated geometric acoustics method". In: *Building and Environment* 160 (2019), p. 106201.
- [112] Yuqi Wang et al. "An efficient GPU-accelerated adaptive mesh refinement framework for high-fidelity compressible reactive flows modeling". In: *Computer Physics Communications* (2025), p. 109870.
- [113] Xiaofei Wu et al. "FluidUrban v1. 0: Enhancing Urban Ventilation and Pollutant Dispersion Modelling with Three-dimensional Dynamic Adaptive Meshes Optimisation". In: *EGUsphere* 2026 (2026), pp. 1–31.
- [114] Zhixiong Xiao et al. "A case study for ray tracing cores: Performance insights with breadth-first search and triangle counting in graphs". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 9.2 (2025), pp. 1–25.

- [115] Zhengzheng Yan, Rongliang Chen, and Xiao-Chuan Cai. "Large eddy simulation of the wind flow in a realistic full-scale urban community with a scalable parallel algorithm". In: *Computer Physics Communications* 270 (2022), p. 108170.
- [116] Furkat Yunus et al. "Efficient prediction of airborne noise propagation in a non-turbulent urban environment using Gaussian beam tracing method". In: *The Journal of the Acoustical Society of America* 153.4 (2023), pp. 2362–2362.
- [117] Sheng Zhang, Lishu Duan, and Hanbo Jiang. "Accelerating Gaussian beam tracing method with dynamic parallelism on graphics processing units". In: *Computer Physics Communications* 315 (2025), p. 109722.



Appendix A

Our conference short paper “A High-Performance Game Engine-GIS-CFD System for Interactive Urban Wind Decision Support” focuses on answer the first subresearch question:

“How can a game engine-GIS-CFD triplet optimize the computational efficiency and usability of urban wind simulators?”

The paper has been accepted by AGILE 2026, and is selected for best paper competition.

A High-Performance Game Engine-GIS-CFD System for Interactive Urban Wind Decision Support

Xuanchen Zhou ¹, Azarakhsh Rafiee ¹, and Peter van Oosterom ¹

¹Department of Architectural Engineering and Technology, Faculty of Architecture and the Built Environment, Delft University of Technology, Delft, Netherlands

Correspondence: Xuanchen Zhou (X.Zhou-6@tudelft.nl)

Abstract. Wind simulation plays a critical role in urban design and planning. Most existing workflows rely on professional and general-purpose computational fluid dynamics (CFD) software. Although powerful and versatile, such tools are not designed specifically for the urban planning process, where interactivity and real-time feedback are highly emphasized. To address this gap, we propose a responsive and specialized decision support system tailored for urban wind simulation. The system enhances the decision-making process by offering an efficient workflow that seamlessly connects data preparation, simulation, and visualization steps. Its core contribution lies in the tight integration of a high-performance CFD model, a game engine, and multi-source geographical information system (GIS) data within a unified environment. The system is built in the Unity game engine to ensure efficiency, user interactivity, scene modifiability and intuitive visualization. The spatial extent and scalability of the system is expanded by incorporating data from multiple GIS sources to represent real-world urban contexts. The wind simulation is accelerated using Unity compute shaders to achieve fast, fully parallelized computation while maintaining physical fidelity. The system also includes NetCDF-based file export and a web viewer. In our testing platform, the system achieves $8.15\times$ acceleration compared to OpenFOAM. The simulation output is also validated through a case study. By combining these elements, we construct a Game Engine-GIS-CFD triplet that serves as a robust decision support system. Its efficiency and usability make it well suited for wind-related urban planning.

Submission Type. software, model, infrastructure

BoK Concepts. [GC2] Spatial simulation modelling, [GC1] Geocomputation and complex systems

Keywords. computational fluid dynamics, decision support system, wind modeling, digital twin

1 Introduction

Urban wind environment significantly impacts various aspects of human society and city life. Local wind patterns directly influence the comfort and safety of pedestrians (Blocken et al., 2012; Hågbo and Giljarhus, 2024). If utilized properly, strong urban wind can serve as a potential energy source (Juan et al., 2022), but it can also pose risks to high-rise buildings (Giangaspero et al., 2022). Moreover, the strategic management of urban wind is essential for mitigating the urban heat island effect (Jing et al., 2021; Taleghani et al., 2014) and enhancing the dispersion of atmospheric pollutants (Murena et al., 2009; Sanchez et al., 2017). Consequently, understanding these wind-environment interactions is vital for numerous applications.

Due to these significant impacts, it is crucial to consider wind effects during the early phase of urban planning (Kaseb and Rahbar, 2022; Kaseb et al., 2020). Generally, the assessment of urban wind effects can rely on multiple different modeling techniques. The diagnostic mass-consistent flow field model (Singh et al., 2008) has been used in predicting wind field using very limited computational resources. However, this acceleration comes at the cost of accuracy loss, as it cannot capture some important wind features, such as channel effect (Gowardhan et al., 2011). By comparison, computational fluid dynamics (CFD) methods provide higher-fidelity predictions by directly solving the governing fluid equations, enabling the resolution of complex flow phenomena between the buildings (Chu and Wang, 2025).

Currently, most urban wind simulations rely heavily on general-purpose CFD software packages. While offering high accuracy and versatility, they are often not specifically optimized for the iterative workflows required in early-stage urban design. They typically have steep learning curves and require substantial user expertise (Hu et al., 2022). A traditional urban CFD workflow involves a labor-intensive setup stage, including building and

terrain model acquisition, format transformation, mesh generation and simulation configuration (Deininger et al., 2020). This complex and fragmented process creates a barrier to utilizing CFD effectively within the fast-paced decision-making processes of urban design (Hu et al., 2022). Integrating CFD workflows with geographical information system (GIS) data sources offers a promising direction to address this challenge, as GIS platforms already maintain detailed and extensive structured urban datasets and provide standardized access to real-world geometry and environmental information.

In this work, we introduce a highly efficient spatial decision support system tailored for wind-aware urban planning. Our study offers three distinct advantages compared to traditional workflows. First, it is integrated with multiple geographical data sources to automatically acquire real-world geometry. Second, it establishes a unified workflow that seamlessly connects model loading, domain initialization, simulation, visualization, and data storage and retrieval, significantly reducing the manual effort required. Finally, it provides a highly efficient, GPU-accelerated wind solver that delivers near real-time performance. All these characteristics make the proposed system especially suited for the interactive exploration of wind environments during the urban planning process.

The remainder of this paper is structured as follows. Section 2 outlines the governing equations underlying the proposed wind simulator. In Section 3, we discuss the methodology used to integrate the game engine, GIS, and CFD simulation into a unified system. Section 4 presents the results, encompassing the wind simulator validation, performance evaluation, and a brief case study to demonstrate our workflow. Finally, Section 5 discusses some key design choices, and Section 6 concludes with a summary of our primary contributions.

2 Background

The numerical simulation of wind flow is fundamentally governed by the Navier-Stokes equations. However, explicitly resolving all turbulent scales (Direct Numerical Simulation) is computationally prohibitive for urban scales (Huo et al., 2021). Thus, urban wind simulations typically rely on two turbulence modeling approaches: Reynolds averaged Navier-Stokes (RANS) and large eddy simulation (LES) (Chu and Wang, 2025). RANS solves for the time-averaged flow field, modeling the effect of all turbulent fluctuations via the Reynolds stress term. In contrast, LES applies a spatial filter to the Navier-Stokes equations, explicitly resolving the large eddies while modeling the smaller, isotropic dissipative structures using a subgrid-scale model. In this work, we adopt incompressible LES as the governing equations, which is obtained by applying a spatial filter on the incompressible

Navier-Stokes equation:

$$\begin{aligned} \nabla \cdot \bar{\mathbf{u}} &= 0, \\ \frac{\partial \bar{\mathbf{u}}}{\partial t} + \nabla \cdot (\bar{\mathbf{u}}\bar{\mathbf{u}}) &= -\nabla \cdot \boldsymbol{\tau} - \frac{1}{\rho} \nabla \bar{p} \\ &+ \nabla \cdot \left(\nu \left(\nabla \bar{\mathbf{u}} + (\nabla \bar{\mathbf{u}})^T \right) \right), \end{aligned} \quad (1)$$

where ρ , $\bar{\mathbf{u}}$, and $\bar{p}$ represent constant density, spatially filtered velocity and spatially filtered pressure, respectively. ν is the kinematic viscosity, $\boldsymbol{\tau}$ is the subgrid-scale stress tensor, defined as $\boldsymbol{\tau} = \bar{\mathbf{u}}\bar{\mathbf{u}} - \bar{\mathbf{u}}\bar{\mathbf{u}}$, which needs to be modelled by a chosen subgrid-scale model (Piomelli, 1999).

Moreover, directly resolving the minute turbulence structures within the viscous sublayer requires an excessively fine grid resolution near the wall and complex local grid refinement techniques. Therefore, we employed a Wall-Modeled LES (WMLES) approach to approximate the shear stress at solid boundaries without refining the mesh down to the wall.

Specifically, we adopted the standard logarithmic law of the wall, as recommended by the Architectural Institute of Japan (AIJ) guidelines for practical wind environment predictions (Tominaga et al., 2008):

$$U^+ = \frac{1}{\kappa} \ln(Ey^+) \quad 30 < y^+ < 200, \quad (2)$$

where E is the wall roughness constant, $U^+ = \frac{U}{u_\tau}$ is the dimensionless velocity, $y^+ = \frac{yu_\tau}{\nu}$ is the dimensionless wall normal distance. u_τ is the friction velocity, and κ is the von Kármán constant.

In urban wind modeling, logarithmic wind profile is often used to estimate the vertical velocity distribution (Tieleman, 2008) as inlet boundary condition:

$$\begin{aligned} U(z) &= \frac{u_{ABL}^*}{\kappa} \ln \left(\frac{z + z_0}{z_0} \right), \\ u_{ABL}^* &= \frac{\kappa U_h}{\ln((h + z_0)/z_0)}, \end{aligned} \quad (3)$$

where $U(z)$ is the inflow value at height z , u_{ABL}^* is the atmospheric boundary layer friction velocity, z_0 is the roughness length, h is the reference height where the reference velocity U_h is obtained.

3 Methods

To establish a highly efficient system for urban wind simulation, a seamless integration of data acquisition, simulation and visualization is essential. This section details the implementation of each module within the proposed decision support system, an overview of the simulation pipeline is shown in Fig. 1.

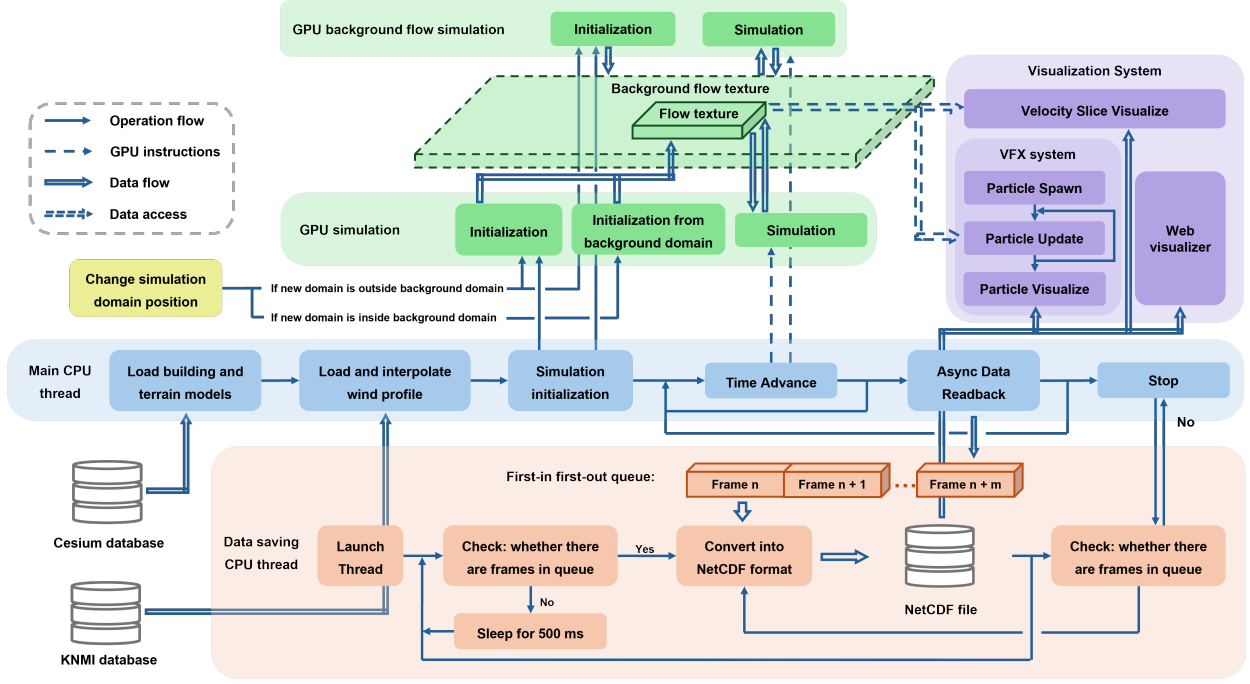


Figure 1. Architecture of the simulator.

3.1 Data Acquisition

Manual geometry acquisition represents a significant workload in traditional urban CFD workflows, which includes building and terrain model acquisition, model loading and format transformation (Deininger et al., 2020). In this work, we create an automated workflow that proceeds directly from region selection to simulation, fully bypassing the tedious model importing and experimental data loading stages.

In order to achieve this, *Cesium for Unity* is used to stream real-world 3D building and terrain models dynamically based on the current viewport (Cesium GS, Inc., 2025). By leveraging the OGC 3D Tiles standard (Open Geospatial Consortium, 2023), the system enables efficient, tile-based data transfer that minimizes memory overhead by preventing unnecessary data loading. Moreover, the wind magnitude and direction are fetched in real-time from the Royal Netherlands Meteorological Institute (KNMI) Data Platform using the Environmental Data Retrieval (EDR) API. Our system queries the k nearest weather stations (where k is a user-defined parameter) relative to the simulation site. The meteorological data from these stations are then interpolated using inverse distance weighting (IDW) to obtain the inlet boundary conditions of simulation.

3.2 Domain Discretization

Traditional CFD workflows often include time-consuming mesh generation and optimization processes, which can take several days to complete (Deininger et al., 2020).

In order to adapt to rapid decision-making processes, an automated voxelization-based method that balances computational cost and accuracy was adopted.

After loading the building and terrain geometries, the simulation domain is discretized into a uniform voxel grid. A ray-casting scanline voxelization algorithm is implemented in our solver. It is accelerated using the Unity Physics `Physics.RaycastAll` API, which provides an integrated R-tree acceleration structure. An average triangle traverse time complexity of $O(\log(n))$ can be achieved, where n is the number of triangles (Chen and Belleman, 2022).

3.3 GPU Accelerated Wind Simulation

Many CFD tools oriented toward urban wind simulation integrate OpenFOAM as their core, such as SimFlow, Eddy3D, and Butterfly (Kastner and Dogan, 2022). While being robust and versatile, their reliance on CPU-based computation limits their efficiency, which is a critical factor in the urban planning process.

In contrast, the development of modern GPUs has provided a solid foundation for hardware acceleration in CFD (Dang et al., 2022). To improve the efficiency of CFD computation, we implemented a Finite Volume Method (FVM) solver entirely within the Unity game engine. Eq. (1) is solved in parallel, together with Eq. (2) to provide the near-wall stress tensor. Smagorinsky model is used to account for the subgrid-scale turbulence. To ensure the solver accuracy while maintaining numerical stability, a blended scheme consisting of 75% central differencing and 25% linear upwind differencing is

applied to the advection term. The central difference discretization scheme is applied to the diffusion term. The computationally intensive fluid dynamics calculations are written in compute shaders to leverage the massive parallelism of modern GPUs. The linear matrix systems are solved using stencil-based iterative methods, avoiding the construction of global matrices which leads to additional memory consumption. This integrated design allows for seamless synchronization with the visualization pipeline while maintaining cross-platform compatibility. The specific performance acceleration achieved by the GPU architecture is discussed in Section 4.2.

3.4 Data Export

For subsequent visualization, validation, and analysis, the simulation's flow field can be exported to the NetCDF-4 format (NSF Unidata Program Center, 2023; Humphries, 2024). This format is well-suited for high-resolution fluid dynamics data due to its self-describing nature and native support for partial file reading, which is highly useful in post processing. Furthermore, to maintain a responsive user experience and a stable frame rate, the data export process is offloaded to a background worker thread. Whenever a save request is triggered, this dedicated asynchronous thread handles the serialization and disk I/O, preventing these expensive operations from blocking the Unity main thread and stalling the real-time simulation.

3.5 User Interaction

To enable urban building configuration, several building models are pre-configured in the framework. During the simulation, these assets can be instantiated, transformed (translated, rotated, scaled), or removed to assess a proposed structure's aerodynamic influence.

Moreover, the user may want to relocate the simulation domain during decision-making. We accelerate this process by maintaining a concurrent large size, low-resolution background flow field. When the primary domain is relocated, its initial state is interpolated from this background field rather than starting from scratch. Consequently, the foreground flow does not need to be reset to $t = 0$, which significantly accelerates the decision-making process.

3.6 Fluid Visualization

Real-time, comprehensive visualization is essential for intuitively conveying wind direction, magnitude, and overall flow dynamics. To support the analysis of simulation results, our framework employs three visualization modules within Unity: planar slices, vector glyphs (arrows), and particle tracing. By integrating the CFD solver directly within the game engine, the visualization modules can directly read data from the simulation textures. This enables a zero-copy rendering

pipeline, avoiding unnecessary data transfer between the simulation and rendering stages. Specifically, the particle tracing module leverages the Unity Visual Effect Graph (VFX Graph) to directly sample the simulated velocity field textures, updating particle positions and orientations on the fly for highly efficient 3D flow visualization. An example is given in Fig. 6. Additionally, the simulator includes a web-based visualization tool for replaying and analysing saved NetCDF datasets, for example the wind field change after moving a building location.

3.7 Data and Software Availability

The developed Game Engine-GIS-CFD system is open-source and freely accessible to facilitate reproducibility. The core framework is available via GitHub at: <https://github.com/puppy2333/UnityWindSimulation>. Additionally, the interactive web visualizer has been deployed on Hugging Face Spaces (<https://xuanchenzhou-windfieldvisualizer.hf.space/>), with its corresponding source code provided at: <https://huggingface.co/spaces/xuanchenzhou/WindFieldVisualizer/tree/main>.

4 Results

This section provides an evaluation of the proposed GPU-accelerated urban wind simulator. Section 4.1 validates the accuracy of our solver using a single high-rise building test case. Section 4.2 compares the efficiency of our solver with OpenFOAM using MPI. Section 4.3 further demonstrates a typical workflow of our solver.

4.1 Validation

The proposed solver is validated using a simplified version of the airflow around an isolated high-rise building benchmark (Liu and Niu, 2016). This configuration is widely recognized as a standard validation experiment within the Architectural Institute of Japan (AIJ) (Tominaga et al., 2008).

Fig. 2 illustrates the computational domain. The building dimensions are $b \times b \times 2b$, where $b = 4m$ is the building width and depth. The computational domain was discretized using stretched grids, with a minimal grid size of $\Delta x = 0.2m$ close to the building. The grid size increases by a constant factor of 1.08 away from the building. The final domain boundaries terminate at the first cell face that strictly meets or exceeds the minimal distance constraints demonstrated in Fig. 2. At the inlet ($x = 0$), a logarithmic wind profile is applied, as detailed in Section 2. The aerodynamic roughness length, z_0 , is set to $0.1m$ to represent typical urban ground roughness. At the outlet, a zero-gradient (Neumann) boundary condition is applied for velocity, alongside a fixed pressure boundary condition. A no-slip boundary condition is enforced at

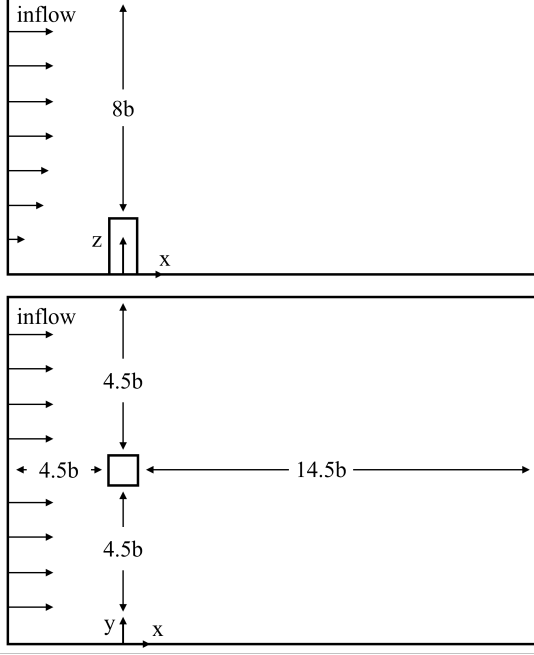


Figure 2. Approximate simulation domain size (up: side view, down: top view).

the ground ($z = 0$). For the top and lateral boundaries, symmetry conditions are applied for both velocity and pressure. The Reynolds number, calculated based on the building height and the reference inlet velocity at a height of $10m$, is 5.333×10^6 . The simulation is conducted using a time step Δt of $0.005s$. It ran for a total of $200s$, yielding a maximum Courant number of 0.6324 , which ensures the Courant-Friedrichs-Lewy (CFL) condition is satisfied throughout.

The results from our solver are compared against a baseline simulation performed in OpenFOAM v2312 (OpenCFD Ltd, 2023). The OpenFOAM setup employs the PISO algorithm, large eddy simulation with the Smagorinsky subgrid-scale model and the log-law wall function. A residual tolerance of 10^{-5} was consistently applied to both the velocity prediction and pressure correction steps across both solvers. However, the native relative residual formulation in OpenFOAM necessitates frequent global reduction operations, which are highly inefficient on GPU architectures. To mitigate this computational bottleneck, we adopted an alternative solution:

$$r = \frac{\|b - Ax\|_2}{N_x N_y N_z}. \quad (4)$$

where r is the calculated residual, N_x , N_y and N_z are the number of cells in x , y and z directions, respectively.

To compare our solver's results with OpenFOAM, mean and root-mean-square (RMS) velocity profiles are extracted along two sample lines, $x/b = 3.5$, $y/b = 5$ and $x/b = 3.5$, $z/b = 0.375$. The sample rate is $20Hz$, with data collected over a time window from $50s$ to $200s$. This

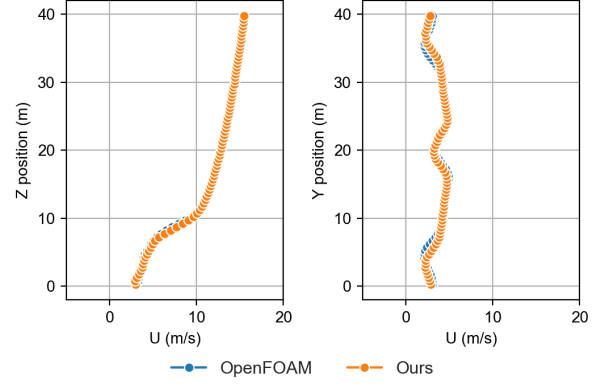


Figure 3. The mean wind velocity distribution around a single building. Left: $x/b = 3.5$, $y/b = 5$, right: $x/b = 3.5$, $z/b = 0.375$.

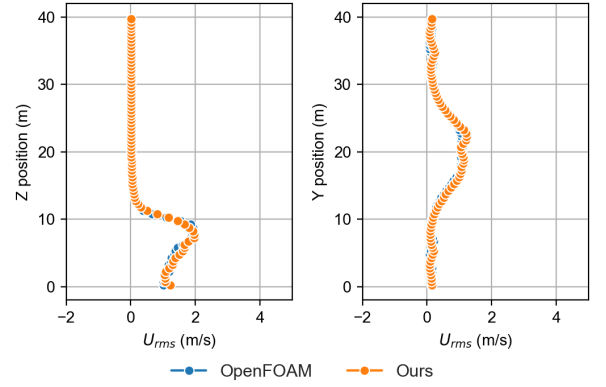


Figure 4. The RMS wind velocity distribution around a single building. Left: $x/b = 3.5$, $y/b = 5$, right: $x/b = 3.5$, $z/b = 0.375$.

collection window is selected to begin after more than five flow-through times, satisfying the criteria outlined in (Okaze et al., 2026). As shown in Fig. 3 and Fig. 4, the validation result demonstrates good overall agreement between the two solvers, confirming the usability of our solver.

4.2 Performance Evaluation

The computational efficiency of our solver is evaluated against OpenFOAM v2312 (OpenCFD Ltd, 2023). The benchmark simulations were conducted on a mobile workstation equipped with an Intel Core Ultra 9 185H CPU and an Nvidia GeForce RTX 4070 Laptop GPU. Due to the hybrid performance-efficiency core architecture of the target CPU and the communication overhead of inter-process messaging, the optimal number of MPI processes for OpenFOAM does not necessarily align with the total physical CPU core count. Consequently, multiple OpenFOAM benchmark configurations were tested and compared with our GPU-based simulator. It was found that 14 processes led to the best performance in our testing platform.

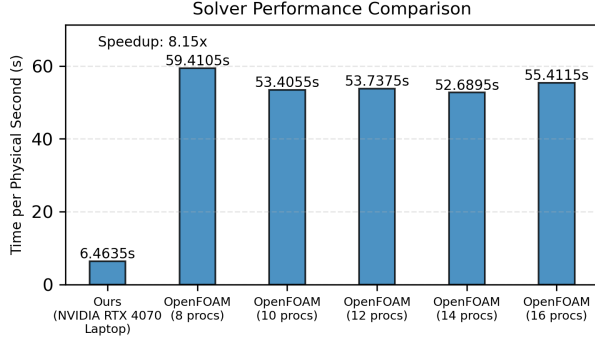


Figure 5. Performance comparison of our simulator and multiple OpenFOAM settings.

As shown in Fig. 5, our simulator achieves approximately $8.15\times$ speedup compared to the most efficient OpenFOAM configuration we tested. This significant speedup is primarily attributed to the efficient parallelization of the CFD algorithms on the GPU.

4.3 Test Case

In order to demonstrate the usability and working pipeline of our simulator, a typical scene including a high-rise building and several low-rise structures is simulated (Fig. 6). The simulation is initialized by defining the domain centroid, dimensions, and grid resolution. For this test case, the scene is chosen at a region in Delft, the Netherlands. The geographical coordinates are (51.9904°N, 4.3766°E), the simulated domain has dimensions of $x = 300m$, $y = 160m$, $z = 80m$, $\Delta x = 1m$. The time step is set to $\Delta t = 0.025s$, yielding a maximum Courant number of 1.02. The inlet boundary condition can be established via two modalities: manual specification and automatic data retrieval. In the manual mode, the user can prescribe specific wind magnitude and direction parameters to test hypothetical scenarios. In the automatic mode, the system retrieves near real-time wind speed and direction from KNMI. The inflow conditions are derived by interpolating data from the four nearest meteorological stations (in this case, two in Rotterdam, one in The Hague, and one in Leiden). The simulation domain is automatically rotated to align with the interpolated wind direction.

Once the boundary conditions are defined, the simulation can be advanced in time. The simulated velocity field can be rendered through several visualization systems, as shown in Fig. 6 (top left, bottom left and right). Users can also instantiate and manipulate several built-in building geometries to observe their aerodynamic influence on the wind field, as shown in Fig. 6 (top right). Furthermore, by coupling the simulation with a nested, coarse-grid background domain (in this case, the domain size is $x = 800m$, $y = 800m$, $z = 80m$, $\Delta x = 4m$), the domain can be relocated during the simulation by interpolating from the large background domain. This architecture

significantly accelerates the flow development compared to a “cold start” for each simulation region relocation.

5 Discussion

In this work, we present a Game Engine-GIS-CFD triplet that enables efficient wind-building interaction simulation. An important design choice was the implementation of a custom CFD solver via Unity compute shaders, rather than coupling our system with external software such as OpenFOAM. While general-purpose CFD software offers broader physical versatility, our compute shader-based approach prioritizes the interactivity essential for early-stage design. As demonstrated in Sections 4.1 and 4.2, our approach achieves better performance while maintaining similar accuracy. Furthermore, zero-copy visualization is achieved by keeping the simulation and visualization pipeline entirely within GPU memory, which allows high-fidelity particle tracing via the VFX Graph with minimal overhead.

Our framework also demonstrates that the triplet architecture is a scalable blueprint for urban digital twins. Beyond wind modeling, this approach can also be adapted to other decision making processes, such as flood or noise modeling, providing powerful tools for early-stage urban development.

6 Conclusion

In this paper, we presented a near real-time, interactive urban wind simulation framework designed to bridge the gap between CFD simulation and iterative urban design. By leveraging GPU-accelerated compute shaders within Unity, we demonstrated the feasibility of obtaining efficient feedback on wind flow patterns in response to building modifications.

The practical workflow of the system was validated through a case study. Our results show that by integrating GIS data acquisition, user manipulation, simulation and visualization into a unified environment, the decision-making process can be executed effectively within a real-time loop, thereby offering critical support during the early stages of urban planning.

Future work will focus on four primary directions. First, we plan to integrate quantitative wind comfort metrics, such as the NEN 8100 and Lawson criteria. These additions can provide planners with actionable, regulatory-compliant insights. Second, we aim to further enhance the solver’s efficiency by implementing adaptive mesh refinement within compute shaders, allowing for the simulation of much larger domains. Third, we plan to extend the solver’s physical models to include scalar transport equations. This will enable the simulation of pollutant dispersion and urban heat island effects, further evolving the framework into a comprehensive

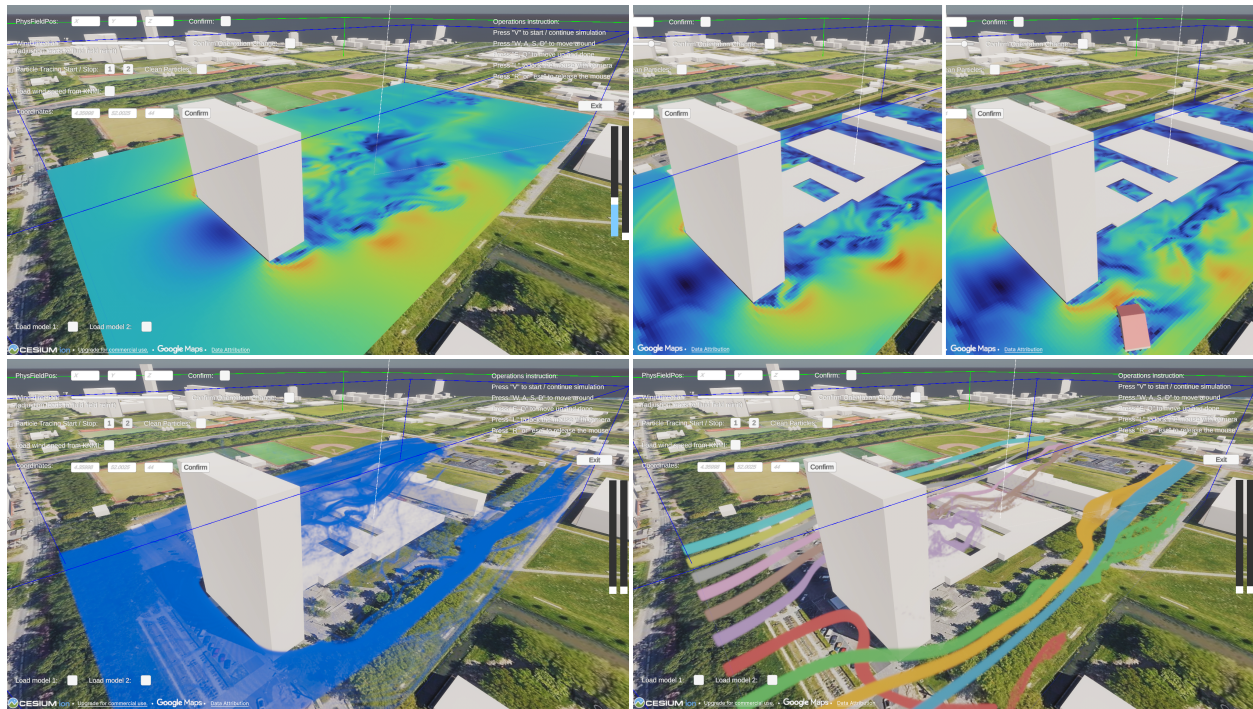


Figure 6. (Top left) The wake generated by a high-rise building. (Top right) Comparison of the fluid field with and without the addition of a small house. (Bottom left and right) Demonstrations of two particle tracing visualization systems.

and versatile tool for climate responsive urban planning. Finally, we aim to further validate the solver by comparing the simulation results against experimental wind tunnel data, and by performing grid convergence tests to evaluate the impact of spatial resolution.

7 Declaration of Generative AI in writing

The authors declare that they have used Generative AI tools in the preparation of this manuscript. Specifically, the AI tools were utilized for language editing, improving grammar, and sentence structure, but not for generating scientific content, research data, or substantive conclusions. All intellectual and creative work, including the analysis and interpretation of data, is original and has been conducted by the authors without AI assistance.

References

- Blocken, B., Janssen, W., and van Hooff, T.: CFD simulation for pedestrian wind comfort and wind safety in urban areas: General decision framework and case study for the Eindhoven University campus, *Environmental Modelling & Software*, 30, 15–34, 2012.
- Cesium GS, Inc.: Cesium for Unity, <https://github.com/CesiumGS/cesium-unity>, accessed: 2026-02-18, 2025.
- Chen, J. and Belleman, R. G.: Measvre: Measurement tools for unity vr applications, in: 2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), pp. 760–761, IEEE, 2022.
- Chu, R. and Wang, K.: CFD in Urban Wind Resource Assessments: A Review, *Energies*, 18, 2626, 2025.
- Dang, G., Liu, S., Guo, T., Duan, J., and Li, X.: Direct numerical simulation of compressible turbulence accelerated by graphics processing unit: An open-source high accuracy accelerated computational fluid dynamic software, *Physics of Fluids*, 34, 2022.
- Deininger, M. E., von der Grün, M., Pieperit, R., Schneider, S., Santhanavanich, T., Coors, V., and Voß, U.: A continuous, semi-automated workflow: From 3D city models with geometric optimization and CFD simulations to visualization of wind in an urban environment, *ISPRS International Journal of Geo-Information*, 9, 657, 2020.
- Giangaspero, G., Amerio, L., Downie, S., Zasso, A., and Vincent, P.: High-order scale-resolving simulations of extreme wind loads on a model high-rise building, *Journal of Wind Engineering and Industrial Aerodynamics*, 230, 105 169, 2022.
- Gowardhan, A. A., Pardyjak, E. R., Senocak, I., and Brown, M. J.: A CFD-based wind solver for an urban fast response transport and dispersion model, *Environmental fluid mechanics*, 11, 439–464, 2011.
- Hågbö, T.-O. and Giljarhus, K. E. T.: Sensitivity of urban morphology and the number of CFD simulated wind directions on pedestrian wind comfort and safety assessments, *Building and Environment*, 253, 111 310, 2024.
- Hu, Y., Xu, F., and Gao, Z.: A comparative study of the simulation accuracy and efficiency for the urban wind environment based on CFD plug-ins integrated into architectural design platforms, *Buildings*, 12, 1487, 2022.
- Humphries, N.: C-sharp-Interface-to-netCDF, <https://github.com/NickHumphries/C-sharp-Interface-to-netCDF>, 2024.

- Huo, H., Chen, F., Geng, X., Tao, J., Liu, Z., Zhang, W., and Leng, P.: Simulation of the urban space thermal environment based on computational fluid dynamics: A comprehensive review, *Sensors*, 21, 6898, 2021.
- Jing, Y., Zhong, H.-Y., Wang, W.-W., He, Y., Zhao, F.-Y., and Li, Y.: Quantitative city ventilation evaluation for urban canopy under heat island circulation without geostrophic winds: Multi-scale CFD model and parametric investigations, *Building and environment*, 196, 107 793, 2021.
- Juan, Y.-H., Rezaeiha, A., Montazeri, H., Blocken, B., Wen, C.-Y., and Yang, A.-S.: CFD assessment of wind energy potential for generic high-rise buildings in close proximity: Impact of building arrangement and height, *Applied Energy*, 321, 119 328, 2022.
- Kaseb, Z. and Rahbar, M.: Towards CFD-based optimization of urban wind conditions: Comparison of Genetic algorithm, Particle Swarm Optimization, and a hybrid algorithm, *Sustainable Cities and Society*, 77, 103 565, 2022.
- Kaseb, Z., Hafezi, M., Tahbaz, M., and Delfani, S.: A framework for pedestrian-level wind conditions improvement in urban areas: CFD simulation and optimization, *Building and Environment*, 184, 107 191, 2020.
- Kastner, P. and Dogan, T.: Eddy3D: A toolkit for decoupled outdoor thermal comfort simulations in urban areas, *Building and Environment*, 212, 108 639, 2022.
- Liu, J. and Niu, J.: CFD simulation of the wind environment around an isolated high-rise building: An evaluation of SRANS, LES and DES models, *Building and Environment*, 96, 91–106, 2016.
- Murena, F., Favale, G., Vardoulakis, S., and Solazzo, E.: Modelling dispersion of traffic pollution in a deep street canyon: Application of CFD and operational models, *Atmospheric Environment*, 43, 2303–2311, 2009.
- NSF Unidata Program Center: Network Common Data Form (NetCDF) Version 4.9.3, <https://doi.org/10.5065/D6H70CW6>, 2023.
- Okaze, T., Kikumoto, H., Ikegaya, N., Nakao, K., Ono, H., Nakajima, K., Imano, M., Hasama, T., Tabata, Y., Kishida, T., et al.: AIJ guidelines on the applications of large-eddy simulation to pedestrian wind environment: Recommendations and validation benchmarks, *Journal of Wind Engineering and Industrial Aerodynamics*, 269, 106 321, 2026.
- Open Geospatial Consortium: 3D Tiles Specification 1.1, OGC Standard 22-025r4, Open Geospatial Consortium, <http://www.opengis.net/doc/cs/3D-Tiles/1.1>, accessed: 2026-02-18, 2023.
- OpenCFD Ltd: OpenFOAM Version v2312, ESI Group, Bracknell, UK, <https://www.openfoam.com/>, 2023.
- Piomelli, U.: Large-eddy simulation: achievements and challenges, *Progress in aerospace sciences*, 35, 335–362, 1999.
- Sanchez, B., Santiago, J. L., Martilli, A., Martin, F., Borge, R., Quaassdorff, C., and de la Paz, D.: Modelling NOX concentrations through CFD-RANS in an urban hot-spot using high resolution traffic emissions and meteorology from a mesoscale model, *Atmospheric Environment*, 163, 155–165, 2017.
- Singh, B., Hansen, B. S., Brown, M. J., and Pardyjak, E. R.: Evaluation of the QUIC-URB fast response urban wind model for a cubical building array and wide building street canyon, *Environmental fluid mechanics*, 8, 281–312, 2008.
- Taleghani, M., Sailor, D. J., Tenpierik, M., and van den Dobbelsteen, A.: Thermal assessment of heat mitigation strategies: The case of Portland State University, Oregon, USA, *Building and Environment*, 73, 138–150, 2014.
- Tieleman, H. W.: Strong wind observations in the atmospheric surface layer, *Journal of wind engineering and industrial aerodynamics*, 96, 41–77, 2008.
- Tominaga, Y., Mochida, A., Yoshie, R., Kataoka, H., Nozu, T., Yoshikawa, M., and Shirasawa, T.: AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings, *Journal of wind engineering and industrial aerodynamics*, 96, 1749–1761, 2008.

B

Appendix B

The Data Management Plan is attached in Appendix B.

Plan Overview

A Data Management Plan created using DMPonline

Title: Game Engine-GIS-Simulation System for Wind and Noise Simulations

Creator: Xuanchen Zhou

Affiliation: Delft University of Technology

Template: TU Delft Data Management Plan template (2025)

Project abstract:

Wind and noise are both key environmental factors that can significantly impact urban livability, making them critical considerations in urban planning. Given the complexity of planning processes, where design decisions must be evaluated and refined iteratively, there is a strong need for accurate and timely feedback on the environmental impact of proposed urban layouts. A (near) real-time simulation system for wind and noise would greatly facilitate this process by enabling more informed and responsive decision-making.

This project focuses on developing a game engine-geographic information system (GIS)-simulation system to support this decision-making process. Through the seamless cooperation of those components, a fast and interactive system can be built. Using information from various web services, an urban digital twin that supports manipulation of scene elements can be constructed inside the game engine. The digital twin is then utilized to perform mesh-based wind simulation and ray tracing-based noise simulation. Through algorithm optimization and GPU acceleration, this project aims to achieve accurate and (near) real-time simulation results. Decision makers can interact with the simulation through multiple interfaces, receiving immediate and informative feedback of the wind and noise levels of current urban building layout.

ID: 192628

Start date: 15-06-2025

End date: 14-06-2029

Last modified: 01-06-2026

Game Engine-GIS-Simulation System for Wind and Noise Simulations

0. Administrative questions

1. Provide the name of the data management support staff consulted during the preparation of this plan and the date of consultation. Please also mention if you consulted any other support staff.

I will consult Dr. Strandberg, the data steward of Faculty of Architecture, during the preparation of this plan.

2. Is TU Delft the lead institution for this project?

- Yes, the only institution involved

1. Data/code description and collection or re-use

3. Provide a general description of the types of data/code you will be working with, including any re-used data/code.

Type of data/code	File format(s)	How will data/code be collected/generated? <i>For re-used data/code: what are the sources and terms of use?</i>	Purpose of processing	Storage location	Who will have access to the data/code?
Self-developed code	C#, HLSL, Python	A software developed for this project.	Perform efficient and interactive urban wind and noise simulation	TU Delft OneDrive, Project data (U:), GitHub	Code in development: the researcher, supervisors. The code will also be open-sourced when the corresponding paper is published.
Code from 3rd party libraries	C#	Open data	Reuse to facilitate software development for the project	TU Delft OneDrive, Project data (U:)	Stored code: the researcher, supervisors. Online code: open source code.
Terrain and buildings models	WMS, WFS, 3D Tiles, CityJSON, obj	Open data	Used in wind and noise modeling	TU Delft OneDrive, Project data (U:)	Stored code: the researcher, supervisors. Online data: open data.
Wind measurement data	EDR API, REST API	Open data (accessed through API)	Used in wind and noise modeling	TU Delft OneDrive, Project data (U:)	Stored code: the researcher, supervisors. Online data: open data.
Road traffic / railway / airline data	WMS, WFS, OGC API, Shapefile, GeoPackage	Open data	Used in noise modeling	TU Delft OneDrive, Project data (U:)	Stored code: the researcher, supervisors. Online data: open data.

II. Storage and backup during the research process

4. How much data/code storage will you require during the project lifetime?

- 250 GB – 5 TB

5. Where will the data/code be stored and backed-up during the project lifetime? (Select all that apply.)

- GitHub/other version control repository (external) – please explain below
- TU Delft OneDrive

- Project Data Storage (U:) drive at TU Delft

III. Data/code documentation

6. What documentation will accompany data/code? (Select all that apply.)

- Software – Usage documentation (README file, docstrings, and in-line comments)
- Metadata – I will adhere to the metadata standards used by the data repository where the data will be shared (see section V)
- Data – README file or other documentation explaining how data are organised

IV. Legal and ethical requirements, code of conducts

7. Does your research involve human subjects or third-party datasets collected from human participants?

If you are working with a human subject(s), you will need to obtain the HREC approval for your research project.

- No

9. Will you work with any other types of confidential or classified data or code as listed below? (Select all that apply and provide additional details below.)

If you are not sure which option to select, ask your Faculty Data Steward for advice.

- No, I will not work with any other types of confidential or classified data/code

10. How will ownership of the data and intellectual property rights to the data be managed?

For projects involving commercially-sensitive research or research involving third parties, seek advice of your [Faculty Contract Manager](#) when answering this question.

This is an internal TUD research project.

V. Data sharing and long term preservation

26. What data/code will be publicly shared?

Please provide a list of data/code you are going to share under 'Additional Information'.

- All data/code underlying published articles/reports/theses
- 1. Code for urban wind simulation.
- 2. Code for urban noise simulation.
- 3. Dataset produced by urban wind and noise simulator.

28. How will you share your research data/code?

Select all that apply and provide additional details below.

- I will share the code via git(lab)/subversion and also create a snapshot in a data repository
- All data/code will be uploaded to 4TU.ResearchData

The code will be shared on GitHub and 4TU.ResearchData. The datasets will be shared using 4TU.ResearchData.

30. How much of your data/code will be shared in a research data repository?

- < 100 GB

31. When will the data/code be shared?

- As soon as corresponding results (papers, theses, reports) are published

32. Under what licence(s) will the data/code be released?

- GPL-3.0+
- CC BY

VI. Data management responsibilities and resources

33. If you leave TU Delft (or are unavailable), who is going to be responsible for the data/code resulting from this project?

The data will be publicly available through GitHub and 4TU.ResearchData.

My supervisors will also be responsible to the data:

- Peter van Oosterom, P.J.M.vanOosterom@tudelft.nl
- Azarakhsh Rafiee, A.Rafiee@tudelft.nl

34. What resources (for example financial and time) will be dedicated to data management and ensuring that data will be FAIR (Findable, Accessible, Interoperable, Re-usable)?

We do not expect to exceed the 1TB data storage provided by 4TU.ResearchData, and therefore there are no additional costs of long term data preservation.

35. Which faculty do you belong to?

- Faculty of Architecture and the Built Environment (ABE)