

A Workflow for Interactive Visualization of Urban CFD Data in Unreal Engine and via Web Streaming

Downloaded from: https://research.chalmers.se, 2026-11-08 10:45 UTC

Citation for the original published paper (version of record):
Eleftheriou, O., Naserentin, V., Pantusheva, M. et al (2026). A Workflow for Interactive Visualization of Urban CFD Data in Unreal Engine and via Web Streaming. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 12(4/W2-2026): 49-56. http://dx.doi.org/10.5194/isprs-annals-XII-4-W2-2026-49-2026

N.B. When citing this work, cite the original published paper.

research.chalmers.se offers the possibility of retrieving research publications produced at Chalmers University of Technology. It covers all kind of research output: articles, dissertations, conference papers, reports etc. since 2004. research.chalmers.se is administrated and maintained by Chalmers Library

(article starts on next page)

A Workflow for Interactive Visualization of Urban CFD Data in Unreal Engine and via Web Streaming

Orfeas Eleftheriou[1], Vasilis Naserentin[1,2], Mariya Pantusheva[3], Dessislava Petrova-Antonova[3], Anders Logg[1]

[1] Department of Mathematical Sciences, Chalmers University of Technology, 412 969 Gothenburg, Sweden – (orfeas, vasilis.naserentin, logg)@chalmers.se
[2] Department of Electrical & Computer Engineering, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
[3] GATE Institute, Sofia 1164, Bulgaria – (mariya.pantusheva, dessislava.petrova)@gate-ai.eu

Keywords: Urban CFD, Interactive Visualization, Unreal Engine, Web Application, Urban Digital Twins, Smart Cities

Abstract

To analyze current and predict future conditions, smart-city planning relies on physics-based simulations to support sustainable design, climate adaptation, and citizen engagement. Yet, the analysis results often remain locked inside specialist tools and static plots that are difficult for non-experts to access and interpret. As part of a broader framework developed for scientific results visualization in an urban context, this paper presents a semi-automated workflow for transforming urban Computational Fluid Dynamics (CFD) outputs into interactive 3D visualizations using Unreal Engine and a web-based streaming application. The feasibility of the workflow is demonstrated through a wind-flow case study of an area in central Sofia, Bulgaria. The proposed approach covers the complete process, from problem formulation and data preparation through simulation execution, results processing, and integration into Unreal Engine. The visualization component is implemented through a modular actor-based architecture that supports both point-based and trajectory-based flow representations, which can be flexibly combined and interactively modified within the scene. Additionally, the visualization system is deployed through Pixel Streaming, providing a practical and viable way to transform complex CFD data into exploratory tools that are accessible to a wide range of urban stakeholders.

1. Introduction

Urbanization is increasing at an unprecedented rate world-wide and as a result, what was once a theoretical aspiration has become a necessary reality to meet the needs of the Smart City. Central to this evolution is the Urban Digital Twin (UDT) - a dynamic virtual replica of the physical environment that integrates diverse data streams to support monitoring, simulation, and planning (Ketzler et al., 2020). Although, UDTs have been successful in integrating static geospatial data and Internet of Things (IoT) feeds into one platform, the integration of high-fidelity physics-based simulations, such as Computational Fluid Dynamics (CFD), introduces its own challenges (Mazzetto, 2024, Lei et al., 2023). Wind flow patterns, urban ventilation, and air pollutant dispersion are critical factors that directly affect pedestrian comfort, building energy efficiency and public health. However, CFD simulation results related to these phenomena, are typically confined to the domain of engineering experts. These datasets are massive, complex, and usually analyzed using specialized post-processing software that generates static 2D/3D plots or offline animations outside of the city context. This creates a communication gap as the stakeholders that can benefit from the analysis results, ranging from city planners and government authorities to real estate developers and citizens, often lack the technical expertise to interpret raw numerical data or vector fields, limiting the impact of simulation insights on decision-making processes. Therefore, a significant need is emerging for visualization tools that are not only physically accurate, but also accessible, interactive and immersive. The rapid advancement of game engines, such as Unreal Engine® (UE; Epic Games, Inc.), offers a solution.With the aid of real-time rendering technologies, it is feasible to convert abstract scientific data to an intuitive visual experience. Nonetheless, the process of migrating data from scientific solvers to graphical engines is a non-trivial task and may require manual conversion or suffer from performance constraints when handling city-scale datasets. This paper presents an interactive visualization pipeline, which was developed to democratize the accessibility to complex urban simulations data. A demonstration of this concept is presented through a case study based on a region of central Sofia, Bulgaria, where wind flow and velocity fields were visualized at true scale. The main contribution of this work lies in the practical application of scientific visualization techniques within a real-time rendering engine, and a stand-alone web streaming application, supported by an automated data processing workflow. The specific objectives of this work are:

- To bridge the gap between expert-level physics simulations and non-expert audiences through high-fidelity, comprehensive and interactive visualization.
- To demonstrate an automated pipeline that converts standard CFD outputs into optimized assets for real-time rendering.
- To validate the accessibility of these results through a web-based streaming architecture, eliminating the need for high-end hardware on the user's side.

The remainder of this paper is structured as follows: Section 2 reviews the state of the art in visualizing physics-based urban data. Section 3 provides an overview of the high-level framework, while Section 4 outlines the simulation prerequisites. Sections 5 through 7 detail the technical implementation within Unreal Engine. Finally, Section 8 describes the web integration, followed by an evaluation and conclusions in Sections 9 and 10.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W2-2026-49-2026 | © Author(s) 2026. CC BY 4.0 License.

2. State of the Art

Urban CFD simulations are widely used for wind flow, ventilation, thermal, and pollutant dispersion analyses in the built environment, and their application is supported by established validation benchmarks and review literature (Tominaga et al., 2008, Toparlar et al., 2017, Pantusheva et al., 2022). However, this methodological maturity at the simulation stage does not always translate to ensuring a broad accessibility of the analysis results and their effective communication to non-technical stakeholders. And these two factors are essential in the context of urban digital twins and smart cities, where simulation outputs are expected to support planning, communication, and operational decision-making in the city, rather than to remain confined to expert analyses (Mitkov et al., 2024, Antoniou et al., 2024, Wu et al., 2023). Mitkov et al. (Mitkov et al., 2024) highlight the role of CFD within UDT ecosystems, identifying the interpretation of CFD results as a major challenge for integrating simulations into planning and decision support. They showcase urban wind simulation results using a combination of Cesium, ArcGIS, and Unreal Engine platforms. Antoniou et al. (Antoniou et al., 2024), similarly, combined realistic 3D rendering with air pollution simulation to inform environmental policy, and Wu et al. (Wu et al., 2023) coupled GIS with CFD for atmospheric dispersion analysis in urban blocks.

From a visualization perspective, with regard to wind flow, different representations reveal different aspects of the same flow field. Post et al. (Post et al., 2003) classified flow visualization techniques into four categories applicable to urban air quality studies: direct methods using glyphs and arrows, texture-based methods for dense flow representation, topology-based methods for identifying flow separation zones, and feature-based methods for turbulent structure visualization. This taxonomy provides a useful reference frame for structuring CFD visualization workflows by integrating perceptual, mathematical, and physical considerations. Tominaga (Tominaga, 2012) applied contour plot visualization for urban ventilation efficiency in Niigata City, introducing the age of air concept as a metric for visualizing pollutant residence time in outdoor urban settings. Deininger et al. (Deininger et al., 2020) compared different visualization techniques for urban CFD wind data, evaluating scalar field visualization, vector field representations, streamlines, and hex-grid formats. The case study they presented showed that streamlines are effective for depicting flow patterns around buildings, and 3D Tiles formats loaded approximately 30% faster than GeoJSON for equivalent datasets. Schneider et al. (Schneider et al., 2020) demonstrated how Open Geospatial Consortium (OGC) standards can enhance web-based accessibility of urban CFD results.

Related and increasingly relevant is the delivery of comprehensive 3D visualizations to end users without requiring local high-performance hardware. Early work by Berger and Cristie (Berger and Cristie, 2015) demonstrated that CFD results could be converted and explored interactively within a game engine at building scale, framing this as a means to bridge the gap between engineers and non-expert stakeholders. Krüger et al. (Krüger et al., 2024) later examined UE specifically as a development environment for scientific immersive visualization, identifying its strengths and practical challenges beyond its original entertainment applications. Somanath et al. (Somanath et al., 2024) demonstrated game-engine-based workflows for city-scale rendering using geospatial data, and (Feng et al., 2024) investigated high-quality Pixel Streaming architectures based on UE and cloud rendering strategies that encode and transmit rendered frames to client browsers in real time. Kim and Suh (Kim and Suh, 2025) proposed an integrated approach to real-time 3D sensor data visualization for digital twin applications, addressing the challenge of making complex spatial data accessible through lightweight client interfaces. However, despite these contributions, efficient cloud-rendered delivery of physics-based simulation results in an urban context remains largely unexplored in the literature.

Related developments in semantic city modeling, urban digital twin data integration and standardization, and stakeholder engaging interfaces have further demonstrated the value of interactive access to interoperable and accessible representations of complex urban information (Gröger and Plümer, 2012, Agugiaro et al., 2018, Wang et al., 2019, White et al., 2021).

Taken together, the literature addresses important parts of the broader urban CFD data visualization workflow, including urban CFD interpretation, comparative evaluation of visualization techniques, interoperable formats, and lightweight delivery of interactive 3D environments. However, these components are typically treated separately and without integration. This work addresses the identified gap by presenting a practical workflow that transforms standard CFD solver outputs and carries them through post-processing, engine integration, real-time visualization, and optional cloud-based deployment for a comprehensive delivery to non-specialist urban stakeholders. The proposed framework and its example implementation are outlined in the following sections.

3. Visualization Framework Overview

The current work focuses on the CFD results visualization in a smart city context. To structure the process from ideation and problem definition to simulation setup and user-facing visualization, the work adopts a modular workflow composed of nine stages, shown in Fig. 1. A brief overview of the structure of this framework is presented below.

1. *Problem definition*: A problem is formulated. Clear objectives are set. Stakeholders' needs and requirements are taken into consideration. Domain, scope, spatial and temporal scales, and variables of interest are specified.

2. *Analysis method selection*: An appropriate analysis approach is selected based on the defined objectives.

3. *Input data acquisition and processing*: Data relevant for the modeled processes are acquired, processed and standardized in a form suitable for further computation and/or visualization.

4. *Simulation/analysis stage*: The selected analysis method is executed to generate results. Outputs can be numerical fields, distributions, or time series, with metadata recorded for reproducibility.

5. *Postprocessing of simulation/analysis results*: In this stage all simulation data are postprocessed and converted into compatible formats with the visualization engine's systems.

6. *Data integration into the visualization engine*: During this stage, the integration of the postprocessed simulation/analysis outputs into the visualization engine is facilitated. Asset pipelines are being set up and the rendering scene is configured.

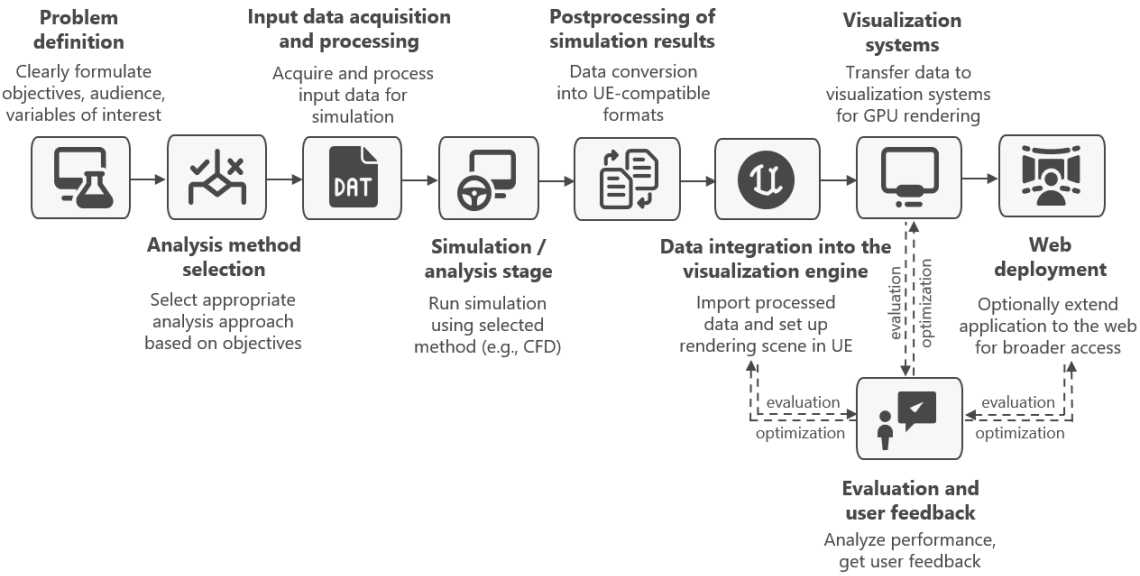


Figure 1. Workflow for smart cities visualizations.

7. *Visualization systems*: Data are transferred to their corresponding visualization systems to enable GPU accelerated visualizations. Depending on the type of data, various visualization systems are developed to provide a cohesive and user friendly experience.

8. *Web deployment*: This is an optional stage which acts as a layer on top of the aforementioned tech stack. Its main purpose is to extend the capabilities of the stand-alone visualization application (e.g. UE) developed from all previous stages and expose it on the Web.

9. *Evaluation and user feedback*: The visualization approach and results (from stages 7 and 8) are evaluated for performance, efficiency, and user satisfaction. This stage allows for iterative refinement of the data processing and visualization stages.

As an illustrative example of the implementation of this generalized framework, the current paper demonstrates the specific use case of integrating CFD simulation results with the real-time rendering capabilities provided by the UE game engine. The outcome is an accessible, comprehensive and interactive visualization system for wind flow analysis that can be used to inform and support decision-making in smart city environments and enhance urban digital twins. Although the workflow is demonstrated here for a wind-flow case study in central Sofia, its structure is sufficiently flexible to support other urban simulation and analysis datasets.

4. Pre-visualization workflow (Stages 1–5)

This section briefly covers the first five stages from the framework, which include all activities related to defining the study's objectives, scope and the data acquisition, processing, and simulation steps required before visualization. Although stages $1-4$ constitute an essential part of the general workflow, they will not be discussed in detail here, as the main focus of the paper is the visualization methodology and its implementation.

4.1 Problem definition

Specific details for the problem formulation stage for the current case study are presented in Table 1.

Category	Details
Objectives	• Interactive exploration and communication of urban wind-flow simulation results • Decision support in urban planning and construction, city management, green initiatives, energy efficiency, urban air quality management, ensuring pedestrian comfort and safety • Citizen awareness and engagement
Domain	• Sofia city central area, Bulgaria • Simulation results cover a radius of 910 m
Geometry scale	• True scale (1:1)
Variables of interest	• Velocity field at pedestrian height • Wind flow pathlines visualizing air movement and recirculation in the city
Target audiences (non-technical users)	• Governmental bodies, local authorities • Urban planners, architects, civil engineers • Citizens and local community stakeholders • Other stakeholders (e.g., real estate developers, local businesses)

Table 1. Case study problem definition

4.2 Analysis method selection

The analysis method chosen for the current case study is CFD simulations utilizing the Reynolds-averaged Navier–Stokes (RANS) equations. This analysis type provides steady time-averaged characteristics of the mean flow at manageable computational cost, which are considered sufficient for the analysis objectives and the target audience needs.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W2-2026-49-2026 | © Author(s) 2026. CC BY 4.0 License.

4.3 Input data acquisition and processing

The city layout was developed using Rhino® 7 and Grasshopper® with the Heron plugin, QuadRemesh, and FME Form®, based on cadastral data, Digital Terrain Model (DTM) with elevation data, aerial orthophotos, publicly available aerial and pedestrian level images. The created geometry was further cleaned and processed in ANSYS® SpaceClaim®, and divided into regions of interest, each capturing a representative Level of Detail (LOD), according to (Kolbe et al., 2021) and the requirements of the study. The cleaned geometry was then discretized in ANSYS Fluent Meshing into finite volumes using a systematic refinement strategy and poly-hexcore meshing. The resulting mesh satisfied standard quality criteria (in terms of mesh statistics and quality metrics), and was deemed suitable for stable and efficient calculations.

Historical environmental data (wind characteristics) from Sofia Airport was collected and analyzed to set relevant boundary and initial conditions for the CFD model.

All data were prepared in formats suitable for import into the simulation tool, namely a `.msh.h5` file for the computational mesh and `.prof` files for the boundary conditions input profiles.

4.4 Simulation/analysis stage

In the current case study, a steady-state RANS CFD simulation using the k–ω turbulence model (Wilcox, 1998) was carried out in ANSYS Fluent® software to analyze the effects of north-northwesterly (NNW) winds passing through the investigated area of central Sofia. The quantities of interest (QoI) presented here are pedestrian-height wind velocity and wind flow pathlines, which can together inform urban planning decisions aimed at improving citizen comfort, street ventilation, and air quality.

Through iterative coordination between the simulation engineer and the visualization developer, suitable export/import formats were identified for the selected quantities of interest. For export from ANSYS Fluent these include: `.csv` format containing spatial coordinates and velocity magnitude values at each node of the computational mesh, and `.fvp` format, containing spatial, temporal, and magnitude components for the wind flow pathlines.

The city-model geometry was also exported via an `.obj` file and shared with the visualization developer, to provide additional context to the simulation results and serve as a visual reference for the scope and LOD of the model.

5. Postprocessing of simulation/analysis results

An intermediate processing layer was developed to transform simulation data into native UE types. Custom Python scripts, Node Converters, and Field View Particles (FVP) were used to parse the data, convert it to the UE coordinate system and generate `.csv` and `.json` files for velocity field and pathlines datasets, respectively. Each original Node entry includes a node number, 3D coordinates, and velocity magnitude. The node number is omitted during postprocessing, as it is irrelevant for the visualization. To ensure uniform spatial alignment across the pipeline, the simulation and visualization data must be synced to a common model space origin.

Direct representation of raw simulation data creates visual gaps between CFD mesh nodes, as shown in Fig. 2. This effect arises because the simulation values are defined at discrete mesh locations, whose spacing is appropriate for the numerical solution but not always for visually continuous rendering at urban scale. Simply increasing the point size in UE fails to resolve this issue, as it introduces overlapping and blurred velocity contours instead, as visible on the bottom row visualizations in Fig. 2.

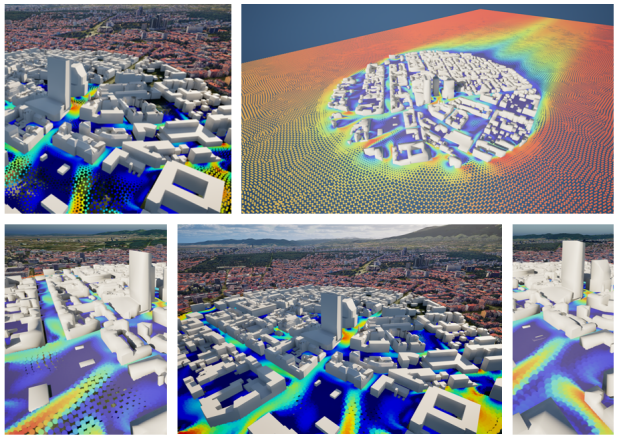


Figure 2. Direct representation of raw CFD data at the computational mesh nodes. Views of wind velocity distribution at pedestrian height (lower values shown in blue, higher - in red).

To improve visual continuity and comprehension, Radial Basis Function (RBF) and piecewise linear interpolation were evaluated for generating intermediate points. This resulted in a denser, smoother scalar field and better velocity magnitude representation, shown in Fig. 3. This initial point-based visualization serves as a baseline prior to the refined visualization system presented later in Section 7.1.

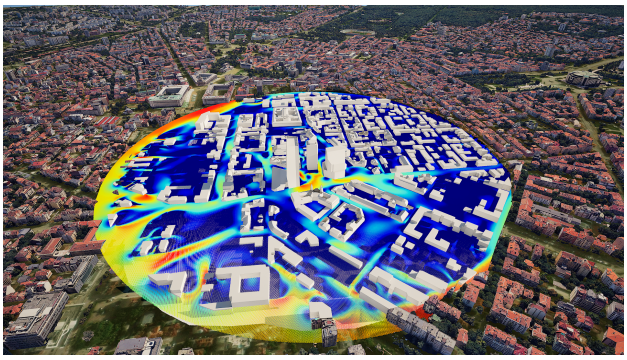


Figure 3. Point-Based Velocity Visualizer without refinement. Visualization of pedestrian-level wind speed.

A similar postprocessing step was applied to the wind-flow simulation results stored in the `.fvp` file, which include pathline tags, variable names, and residence times. The converter discards this non-visual data, keeping only the number of entries, 3D coordinates, time, and velocity magnitude. It then transforms this data to the UE coordinate system and outputs a directly importable (to the engine) `.json` file. Because Velocity and Location arrays are index-matched, UE's built-in parser can automatically detect fields by name and type. This generates structured simulation data tables within the engine, which provide the basis for the point-based and trajectory-based visualization systems described in the following sections.

Overall, for the selected case study, the point-based velocity

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W2-2026-49-2026 | © Author(s) 2026. CC BY 4.0 License.

data at a horizontal plane at pedestrian height consists of 460037 nodes, while the number of generated streamlines equals 845.

6. Data integration into the visualization engine

A modular actor-based framework was developed to support real-time rendering of simulation data. In the UE context, actors encapsulate rendering logic and components. The system uses the Blueprint visual scripting to enable rapid iteration as simulation parameters and data formats evolve, and the architecture employs a Strategy Pattern, allowing actors to use unique parsing and visualization approaches through a shared interface, enhancing modularity. Two specialized actors were implemented: the **Point-Based Visualization Actor** for point-based velocity data and the **Streamline Visualization Actor** for flow trajectories. Both actors implement a common interface with four core operations. *Clean up Data* resets the actor's state to prevent artifacts, *Parse Data* casts data tables into predefined schemas for the particle renderer, *Activate Visualization* triggers the visualization logic, and *Deactivate Visualization* deactivates visualizations without unloading data. Deactivation allows the user to isolate specific datasets, reducing GPU load and visual clutter.

Decoupling parsing and rendering into distinct components ensures a scalable and flexible system that easily integrates new simulation outputs and formats. This modular design facilitates future extensions with minimal changes to the existing codebase.

7. Visualization systems in Unreal Engine

As the model geometry of the city layout was packaged into an `.obj` file, Unreal Engine is able to parse it via the integrated Interchange Framework. This framework implements format agnostic importing of assets. Visualization of `.obj` files is supported natively in UE, hence visualizing the model geometry did not require any additional custom processing.

The **Point-Based Visualization** and **Streamline Visualization Actors** use tailored Niagara System Components to render the simulation data. Niagara is UE's Visual Effects (VFX) System and provides GPU-accelerated rendering suitable for large particle-based datasets. By combining custom materials and simulation parameters the system achieves high-fidelity, GPU-accelerated visualizations that scale efficiently.

7.1 Point-Based Velocity Visualizer

Among the particle systems developed for the framework, the **Point-Based Velocity Visualizer** is presented first. The system serves as a foundational component in the pipeline due to its relatively simple logic and implementation. It comprises a single emitter, responsible for spawning sprites in the scene where each sprite corresponds to an individual data entry from the simulation output. This system utilizes three primary parameters, a *Spawn Rate* which uses a gradual spawning mechanism to prevent performance degradation from thousands of points, *Uniform Sprite Size*, a global variable ensuring consistent scaling and visual coherence across visualizations and a *Point Data array*, a 1D array of 4D vectors generated during the parsing process. Each entry contains a 3D location and a fourth component for velocity magnitude.

By design, Niagara emitters operate through a sequence of execution stages to prepare parameters for the final rendering. Once

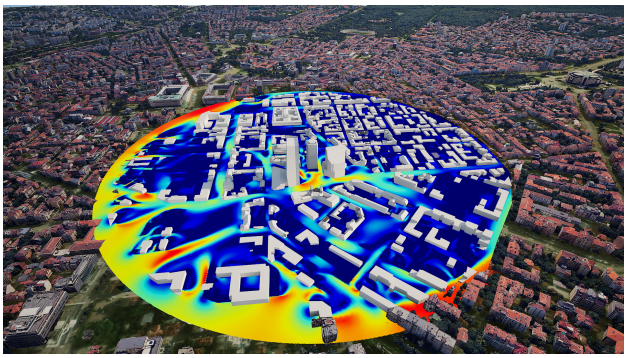


Figure 4. Refined Point-Based Velocity Visualizer. Visualization of pedestrian-level wind speed

the aforementioned parameters are passed into the particle system, the emitter performs the following operations:

1. Each particle is initialized with a defined lifetime, i.e. the duration for which the sprite remains rendered in the scene.
2. The index of the particle is mapped to a corresponding entry in the *Point Data* array.
3. The particle is positioned at its respective 3D location in world space based on the data entry.
4. A dynamic material parameter is set according to the velocity magnitude value. This parameter is later utilized within the material shader to determine the final sprite color via an assigned color map.
5. The particle size is then set according to the global uniform sprite size value.

Through this setup a high-performance, data-driven refined point-based visualization is achieved, as shown in Fig. 4. It uses GPU-accelerated rendering, effectively demonstrating the integration of simulation data into real-time visualization workflows within Unreal Engine. Both performance and visualization clarity are improved in comparison to the unrefined point-based representation discussed in Section 5 and demonstrated in Fig. 3.

7.2 Streamlines Visualizer

To visualize streamline data, the **Streamlines Visualizer** particle system was developed. The system consists of a single emitter, responsible for spawning and animating sprites along the path of individual streamlines. As with the point-based system, the simulation data consists of discrete 3D positions paired with their corresponding velocity magnitudes. In order to enable animation along the paths of the streamlines, the system requires access to both position and velocity data for each point, as well as indexing information to distinguish between individual streamlines. Although originally designed with custom structs, a bug in UE 5.4 necessitated storing streamline data in four separate flat arrays: 1) 3D coordinate vectors for all points; 2) index-aligned velocity magnitudes; 3) streamline starting indices, and 4) streamline ending indices.

This layout uses the shared *Spawn Rate* and *Uniform Sprite Size* parameters, allowing the emitter to procedurally reconstruct and animate streamlines without complex data types. During initialization, each sprite is assigned to a particular streamline. The corresponding start and end indices are then used to identify

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W2-2026-49-2026 | © Author(s) 2026. CC BY 4.0 License.

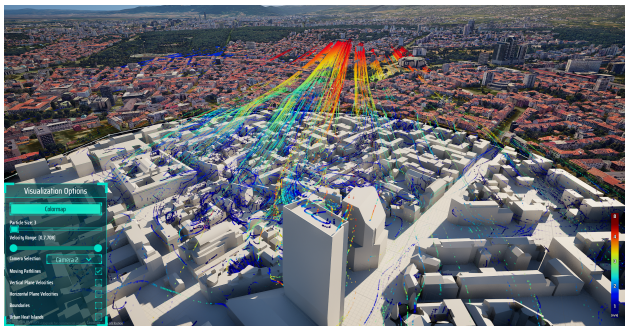


Figure 5. Streamlines Visualizer point of view 1. Wind flow paths representation

the ordered sequence of points defining that streamline and to retrieve the associated velocity values. The sprite motion is generated through interpolation between consecutive points along the assigned path. When a sprite reaches the final point of a streamline, it is returned to the starting location, creating a continuous looping animation. To color the sprites based on velocity, the same material system used in the **Point-Based Velocity Visualizer** is applied. The primary difference lies in the dynamic update of the velocity parameter as the sprite moves along the streamline. Additionally, a consistent vector offset is applied to each sprite's position to produce a visually cohesive "tube-like" effect, enhancing the perception of streamline geometry. The visualization resulting from this methodology for the current case study is presented in Fig. 5.

8. Web deployment

Since Unreal Engine 5.4 lacks native web or WebGPU export, Pixel Streaming was integrated to provide web-based access. This allows GPU-intensive applications to run on a remote server and stream to client browsers, eliminating the need for local high-end hardware or software installation.

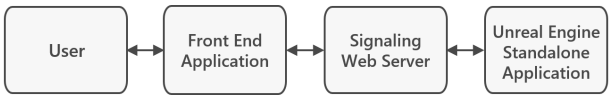


Figure 6. Streaming pipeline

The architecture includes a TypeScript/NodeJS frontend web application and a Signalling Web Server, as shown in Fig. 6. The frontend provides the user interface, allowing direct integration with the UE application. The Signalling Web Server coordinates data exchange between the frontend and the UE instance, routing interactions and streaming video frames. It uses configurable arguments to manage different simulation entry points. Once deployed, the frontend connects to the server to establish a direct, interactive communication channel with the UE runtime.

9. Evaluation and user feedback

This section evaluates the performance of the stand-alone Unreal Engine application together with initial user feedback.

9.1 Performance Assessment

The Unreal Engine application is mostly GPU-bound due to the nature of the visualization systems as all the visualizations are handed off to the GPU. To evaluate responsiveness under typical usage, the application was profiled under two different hardware configurations in three different scenarios 1) all visualizations disabled (Off), to establish a baseline for the system, 2) point-based Velocity Plane visualization enabled, and 3) Streamlines visualization enabled. The measured CPU and GPU frame times are reported in Table 2.

Configuration	Visualization	CPU time [ms]	GPU time [ms]
Ryzen 3950X	Off (Baseline)	9	13
& GTX 1080Ti	Velocity Plane	10	17
(desktop)	Streamlines	10	14.5
Intel i7-11800H	Off (Baseline)	12	16
& RTX 3080	Velocity Plane	13	24
(laptop)	Streamlines	13	17

Table 2. CPU and GPU frame times for two hardware configurations across the three visualization scenarios.

The frame time of the GPU for both implementations ranged from approximately 13 ms to 24 ms (corresponding to $\approx$ 41 FPS to 76 FPS) while the observed video bitrate ranged between 50-65 Mbps with measured end-to-end latency remaining within 80-90ms for a single user. These values fall within commonly cited ranges for interactive real-time visual applications. In particular, Li et al. (Li et al., 2015) characterize frame rates above 26 FPS as suitable for smooth interaction, while Google's RAIL performance model associates smooth animation with an approximate 16 ms frame budget and responsive interaction with latencies below 100 ms (Google Chrome Developers, n.d.). The behavior of the developed visualization application satisfy both criteria. No stuttering or input lag were perceived during navigation, interaction, or switching between visualization modes. The CPU times didn't show much variation across the three scenarios and confirmed that the application is primarily bound by the GPU and the performance scales, as expected, with the visualization complexity.

In summary, the presented results confirm that the proposed visualization system operates well and can support interactive exploration of CFD data sets on both tested configurations, representing mid-to-high performance consumer hardware, and does not require specialized workstation or HPC systems. This supports the technical feasibility of the proposed approach for broader deployment beyond specialist computing environments.

9.2 User Perception and Feedback

A primary goal of the proposed visualization pipeline is to improve the accessibility and interpretability of CFD results for non-expert audiences, as reflected in the user requirements defined in the initial stages of the workflow (Table 1). These requirements were developed based on initial discussions among the visualization, simulation, and digital twin experts, who provided the input to represent the interests of all stakeholders in the project including municipal authorities, urban planners, and the general public. Based on their input, the types of visualization tools, results scope, interaction mechanisms, and usability features were determined.

Following the first functional prototype, qualitative feedback was collected from the simulation experts and the city digital twin team. The need for providing a legend, different camera viewpoints, and free camera movement was identified, so that users can conduct an exploratory analysis of the local flow characteristics and zoom in/out to see details of the flow behavior at any location in the model space. In response to this input, all

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W2-2026-49-2026 | © Author(s) 2026. CC BY 4.0 License.

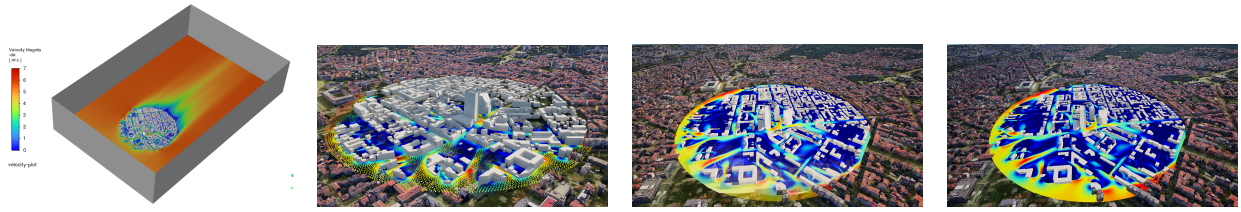


Figure 7. Visualizations of the velocity field in the investigated region via (left to right): 1/ ANSYS Fluent; 2/ Visualization of raw point data in UE; 3/ Unrefined Point-Based Velocity Visualizer; 4/ Refined Point-Based Velocity Visualizer

features were implemented in subsequent versions. Next, to improve the usability of the application and to reduce the learning curve of using the visualizations systems a user interface (UI) was introduced that allows:

- enabling or disabling multiple visualization systems;
- adjusting the particle size;
- dynamically changing threshold values for the visualized parameter;
- switching between different colormaps.

The availability of multiple colormaps, can greatly enhance accessibility, as certain palettes are more suitable for users with color vision deficiencies or other visual impairments. These additional capabilities improved the ease of navigating in the application and supported more flexible inspection of the CFD results.

To further examine how visualization design impacts the perceptual understanding of CFD results, the same wind velocity field is compared through four different representations shown in Fig. 7: (1) a standard contour plot native to Fluent, (2) a direct visualization in UE of the raw point-data output from each node of the computational mesh within the investigated region, (3) a visualization through the Point-based Visualizer (discussed in Section 5), and its refined version (4) (Section 7.1). Informal feedback from both expert and non-expert viewers showed clear differences in how easily each representation could be interpreted. The Fluent plot Fig. 7 (1) was generally perceived as the most difficult to comprehend because it lacked city context, required prior CFD knowledge to interpret the color gradients, and also depends on licensed-tier access. The raw point-data view Fig. 7 (2) conveyed the underlying mesh more transparently, but its discrete appearance can obscure continuous flow patterns and key acceleration/recirculation zones. The refined visualizer on Fig. 7 (4) was consistently rated most intuitive. Viewers linked denser sampling and smooth color transitions to better spatial coherence and an easier read of local wind behavior against the surrounding urban geometry.

Although the intended end-users did not interact directly with the UE model or the standalone web application during this development stage, several stakeholders, including citizens and municipality representatives, were shown demonstrations of the visualizations and provided positive informal feedback regarding clarity, realism, and ease of interpretation of the CFD results. Overall, this suggests that the proposed approach may improve the interpretability of CFD results relative to traditional outputs. However, these observations remain preliminary and should be complemented by structured user evaluation in future work.

9.3 Supplementary Material

Demonstration videos of the interactive visualization are available at `https://www.youtube.com/watch?v=HDkAcKk5WdA` and `https://www.youtube.com/watch?v=CArGLPd4nis`. Additional information about the project can be found at `https://city.gate-ai.eu/projects/Pedestrian-Wind-Comfort.html`.

10. Conclusions and Future Work

This paper presented an end-to-end workflow for transforming physics-based urban simulation results into interactive 3D visualizations using Unreal Engine and web-based streaming technologies. By integrating CFD output with GPU accelerated particle systems and a modular actor-driven architecture, the proposed pipeline demonstrated that simulation data can be rendered and explored in real time, both locally and through web browsers. The workflow maintains a clear separation between simulation, post-processing, data integration, and visualization stages, which makes it adaptable to a wide range of scientific domains and urban digital twin applications. The presented implementation demonstrated that the approach is technically feasible, performant, and can provide interactive exploration of CFD wind flow simulation results to diverse stakeholders, and thus support informed decision making.

Future work developments will focus on extending the framework toward a more scalable and production-ready architecture suitable for continuous deployment within operational digital-twin platforms. Specifically:

- automation and continuous integration/development (CI/CD) workflows could support containerized deployment and more efficient publication of new datasets and visualization updates to the web platform;
- support for additional simulation outputs, such as temperature fields, pollutant dispersion, and pedestrian-density data, as well as real-world data streams, could extend the workflow to a broader range of urban digital-twin use cases and applications in different industries;
- improved user interfaces and introducing additional interaction modes, time-varying playback, annotations or multi-user, as well as domain-specific overlays, could further enhance data interpretability and usability.

Overall, the presented workflow demonstrates a robust foundation for interactive scientific visualization in urban applications and its modularity ensures that it can evolve toward increasingly realistic, data-rich and operational digital-twin environments.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W2-2026-49-2026 | © Author(s) 2026. CC BY 4.0 License.

11. Acknowledgements

This research is part of the GATE project funded by the Horizon 2020 WIDESPREAD-2018-2020 TEAMING Phase 2 programme under grant agreement no. 857155 and the programme "Research, Innovation and Digitalization for Smart Transformation" 2021-2027 (PRIDST) under grant agreement no. BG16RFPR002-1.014-0010-C01, the DS4SSCC–DEP is funded by the European Union Digital Europe Work Programme 2021-2022 under Grant Agreement No. 101123342 and the FLEdge project, F-DUT-2022-0337, funded by the European partnership Driving Urban Transitions (DUT) under agreement no. KP-06-D002/5 with the Bulgarian Scientific Fund and agreement no. P2024-00145 with Swedish Energy Agency (SWEA). The authors gratefully acknowledge the support of the Digital Twin Cities Centre, supported by Sweden's Innovation Agency Vinnova under grant agreement no. 2024-03904.

References

Agugiaro, G., Benner, J., Cipriano, P., Nouvel, R., 2018. The Energy Application Domain Extension for CityGML: Enhancing interoperability for urban energy simulations. *Open Geospatial Data, Software and Standards*, 3, 2.

Antoniou, A., Ioannidis, G., Ntziachristos, L., 2024. Realistic simulation of air pollution in an urban area to promote environmental policies. *Environmental Modelling & Software*, 172, 105918.

Berger, M., Cristie, V., 2015. CFD post-processing in Unity3D. *Procedia Computer Science*, 51, 2913–2922.

Deininger, M. E., von der Grün, M., Piepereit, R., Schneider, S., Santhanavanich, T., Coors, V., Voß, U., 2020. 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(11), 657.

Feng, H., Lu, H., Zhang, F., Li, Z., 2024. Application and research of high quality pixel streaming architecture based on unreal engine. *Proceedings of the 3rd International Conference on Computer, Artificial Intelligence and Control Engineering*, 17–21.

Google Chrome Developers, n.d. Rail: A user-centric performance model. `https://web.dev/rail/`. Accessed: 2025-12-01.

Gröger, G., Plümer, L., 2012. CityGML – Interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71, 12–33.

Ketzler, B., Naserentin, V., Latino, F., Zangelidis, C., Thuvander, L., Logg, A., 2020. Digital Twins for Cities: A State of the Art Review. *Built Environment*, 46(4), 547–573.

Kim, H., Suh, H., 2025. An Integrated Approach to Real-Time 3D Sensor Data Visualization for Digital Twin Applications. *Electronics*, 14(15), 2938.

Kolbe, T. H., Kutzner, T., Smyth, C. S., Nagel, C., Roensdorf, C., Heaze, C. (eds), 2021. OGC City Geography Markup Language (CityGML) Part 1: Conceptual Model*, OGC Standard 20-010, Version 3.0*. Open Geospatial Consortium.

Krüger, M., Gilbert, D., Kuhlen, T. W., Gerrits, T., 2024. Game engines for immersive visualization: Using unreal engine beyond entertainment. *PRESENCE: Virtual and Augmented Reality*, 33, 31–55.

Lei, B., Janssen, P., Stoter, J., Biljecki, F., 2023. Challenges of urban digital twins: A systematic review and a Delphi expert survey. *Automation in Construction*, 147, 104716.

Li, Q., Xu, P., Qu, H., 2015. FPSSeer: Visual analysis of game frame rate data. *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, 73–80.

Mazzetto, S., 2024. A Review of Urban Digital Twins Integration, Challenges, and Future Directions in Smart City Development. *Sustainability*, 16(19), 8337.

Mitkov, R., Pantusheva, M., Petrova-Antonova, D., Naserentin, V., Logg, A., 2024. The role of computational fluid dynamics within city digital twins: Opportunities and challenges. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W4-2024, 131–136.

Pantusheva, M., Mitkov, R., Hristov, P. O., Petrova-Antonova, D., 2022. Air pollution dispersion modelling in urban environment using CFD: A systematic review. *Atmosphere*, 13(10), 1640.

Post, F. H., Vrolijk, B., Hauser, H., Laramee, R. S., Doleisch, H., 2003. The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum*, 22(4), 775–792.

Schneider, S., Santhanavanich, T., Koukofikis, A., Coors, V., 2020. Exploring schemes for visualizing urban wind fields based on CFD simulations by employing OGC standards. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VI-4/W2-2020, 157–163.

Somanath, S., Naserentin, V., Eleftheriou, O., Sjölie, D., Wästberg, B., Logg, A., 2024. Towards Urban Digital Twins: A Workflow for Procedural Visualization Using Geospatial Data. *Remote Sensing*, 16(11), 1939.

Tominaga, Y., 2012. Visualization of city breathability based on CFD technique: Case study for urban blocks in Niigata City. *Journal of Visualization*, 15(3), 269–276.

Tominaga, Y., Mochida, A., Yoshie, R., Kataoka, H., Nozu, T., Yoshikawa, M., Shirasawa, T., 2008. AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, 96(10–11), 1749–1761.

Toparlar, Y., Blocken, B., Maiheu, B., van Heijst, G. J. F., 2017. A review on the CFD analysis of urban microclimate. *Renewable and Sustainable Energy Reviews*, 80, 1613–1640.

Wang, H., Pan, Y., Luo, X., 2019. Integration of BIM and GIS in sustainable built environment: A review and bibliometric analysis. *Automation in Construction*, 103, 41–52.

White, G., Zink, A., Codecá, L., Clarke, S., 2021. A digital twin smart city for citizen feedback. *Cities*, 110, 103064.

Wilcox, D., 1998. *Turbulence Modeling for CFD*. DCW Industries.

Wu, Q., Wang, Y., Sun, H., Lin, H., Zhao, Z., 2023. A system coupled GIS and CFD for atmospheric pollution dispersion simulation in urban blocks. *Atmosphere*, 14(5), 832.