




AirFogSim: A Python Package for Benchmarking Collaborative Intelligence in Low-Altitude Vehicular Fog Computing

Zhiwei Wei ¹, Bing Li ², and Rongqing Zhang ²✉

¹ Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, China ² School of Computer Science and Technology, Tongji University, China ✉ Corresponding author

DOI: [10.21105/joss.08267](https://doi.org/10.21105/joss.08267)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Patrick Diehl](#)  

Reviewers:

- [@mrsonandrade](#)
- [@mengdayayyt](#)
- [@hghk-bit](#)

Submitted: 20 April 2025

Published: 01 July 2025

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

AirFogSim is an Agent-Based Modeling (ABM) simulation package designed specifically for benchmarking collaborative intelligence in Low-Altitude Vehicular Fog Computing (LAVFC) systems. It provides a lightweight yet comprehensive Python-based simulation framework for modeling the complex dynamics, resource constraints, and collaborative intelligence strategies within these LAVFC systems, facilitating research and development in this emerging field.

Statement of need

The proliferation of Unmanned Aerial Vehicles (UAVs) is transforming low-altitude airspace, enabling the *Low-Altitude Economy* ([Huang et al., 2024](#)) with diverse applications in logistics, surveillance, and advanced air mobility. Managing this complex and dynamic environment effectively demands intelligent, scalable, and real-time solutions ([Wei et al., 2024](#)). *Low-Altitude Vehicular Fog Computing* (LAVFC) emerges as a promising paradigm, integrating UAVs with vehicular networks and fog computing infrastructures to facilitate efficient, collaborative operations ([Hu et al., 2021](#)). However, evaluating the performance and feasibility of diverse LAVFC strategies is challenging due to the complex interplay of mobility, communication, computation, and resource constraints, thus necessitating robust and flexible simulation tools.

Existing platforms like iFogSim ([Gupta et al., 2017](#); [Mahmud et al., 2022](#)) and EdgeCloudSim ([Sonmez et al., 2017](#)) excel in modeling computation and basic energy aspects but typically lack realistic mobility and detailed communication channel models crucial for dynamic air-ground scenarios. Conversely, network-centric simulators such as Veins ([Sommer et al., 2019](#)) and FogNetSim++ ([Qayyum et al., 2018](#)) provide high-fidelity communication and traffic modeling, often leveraging OMNeT++ and SUMO ([Krajzewicz et al., 2012](#)), yet generally require substantial extensions to incorporate sophisticated computation task management or energy constraints. Other specialized tools may focus primarily on vehicular fog contexts like VFogSim ([Akgül et al., 2023](#)), whose advanced channel modeling may depend on proprietary software like WinProp, or concentrate on UAV operations as seen in Anylogic's simulator ([Mohebbi & Murali, 2022](#)), implemented using the commercial AnyLogic platform, or MARSIM ([Kong et al., 2023](#)) using ROS, whose availability may be restricted.

Platform	Mobility Modeling	Computation Modeling	Communication Modeling	Energy Modeling	Environment/Topology	Open Source (License)	Language	Key Dependencies	Primary Use Case
iFogSim	No / Yes (Fog Node, v2)	Yes (Tasks, VMs)	Yes (Abstract Delay)	Yes	No	Yes (GPLv3)	Java	CloudSim	IoT/Fog Resource Mgmt (Static)
EdgeCloudSim	Yes (User, WiFi Handoff)	Yes (Tasks, VMs)	Yes (Link Delay, WLAN/WAN)	No	No	Yes (GPLv3)	Java	CloudSim	MEC/User Mobility
FogNetSim++	Yes (Vehicular, Custom)	Yes (Tasks)	Yes (Packet-level, Wireless)	No	No	Yes (Academic)	C++	OMNeT++	VFC/Fog (Network Detail Focus)
Veins	Yes (Vehicular, SUMO Traffic)	No	Yes (Packet-level, V2X Detail)	No	Yes (Road Network)	Yes (GPLv2)	C++	OMNeT++, SUMO	V2X Protocol Evaluation
VFogSim	Yes (Vehicular, SUMO/Trace)	Yes (Tasks)	Yes (Link-level, V2N, Channel)	Yes	Yes (Road Network)	Limited	C++	SUMO, WinProp	Data-driven VFC Analysis
MARSIM	Yes (UAV, Physics-based)	No	No	No	No (or Abstract 3D)	Limited	C++	ROS	UAV Sensing Simulation (Small Scale)
ABS Model (Akbari et al.)	Yes (Vehicular, ABM)	No	Yes (Abstract V2X for CAVs)	No	Yes (Road Network)	Limited	Java	AnyLogic	CAV Impact on Traffic Flow Concepts
AirFogSim	Yes (Vehic. (SUMO)+UAV(Config.))	Yes (Tasks, Offload)	Yes (Simplified PHY, Air-Ground)	Yes	Yes (Road + 3D Space)	Yes (Apache 2.0)	Python	Optional DataLoader	Integrated LAVFC Benchmarking

Figure 1: Comparison of other simulation tools

AirFogSim distinguishes itself by offering an integrated, open-source framework specifically designed to bridge these gaps for the burgeoning LAVFC domain (Wei et al., 2024). Developed in Python for enhanced flexibility and accessibility, it employs a unified workflow-agent-task architecture tailored for diverse LAVFC applications. Furthermore, to bolster realism beyond pure simulation, AirFogSim uniquely provides standardized DataLoader and DataIntegration interfaces, facilitating the seamless incorporation of real-world datasets, such as traffic data from SUMO (Krajewicz et al., 2012) or meteorological data via OpenWeatherMap API.

AirFogSim Fundamentals

Key concepts in AirFogSim include:

- **Environment:** Orchestrates simulation time, manages agent/resource registries, and facilitates event communication via the EventRegistry. It extends SimPy's Environment and serves as the central hub for all simulation entities and managers.
- **Agent:** Represents active entities (UAVs, base stations, edge nodes). Each has an ID, internal state (position, battery), attached components, and can interact with workflows through the environment's workflow manager.
- **Component:** Encapsulates agent capabilities (e.g., MoveToComponent, CPUComponent). Components monitor agent state, generate performance metrics, and execute relevant tasks.
- **Task:** An atomic unit of work (e.g., MoveToTask, ImageProcessingTask) executed by a component. Tasks consume metrics, calculate progress, potentially modify agent state (PRODUCED_STATES), and adhere to priorities.
- **Workflow:** Defines higher-level processes or mission plans (e.g., InspectionWorkflow, ChargingWorkflow) involving multiple tasks. Uses a WorkflowStatusMachine to manage states and transitions triggered by time, events, or agent state changes.
- **DataProvider:** Integrates external data sources (e.g., weather, traffic) into the simulation environment, enabling more realistic scenarios.
- **EventRegistry:** Central publish-subscribe system for decoupled communication between simulation entities.
- **Resource / ResourceManager:** Models limited shared assets (e.g., charging spots, spectrum) managed by specific managers.

Environment manages the simulation and contains various managers; Agent represents entities with state and components; Component provides capabilities and executes tasks; Task represents

atomic units of work; and Workflow defines higher-level processes using state machines.

Developing AirFogSim: Exploring Custom Extensions

AirFogSim's object-oriented design facilitates extension through inheritance.

1. **Custom Agent:** Inherit from `core.agent.Agent` (or subclasses like `DroneAgent`). Define custom `_state_templates` in `AgentMeta` subclasses for unique states.
2. **Custom Component:** Inherit from `core.component.Component`. **Determine** capabilities based on `agent.state` via `MONITORED_STATES`. Implement `_calculate_performance_metrics` to generate `PRODUCED_METRICS`.
3. **Custom Task:** Inherit from `core.task.Task`. Define class attributes `NECESSARY_METRICS` and `PRODUCED_STATES`. Implement the core logic within the `execute()` SimPy generator function, yielding delays, handling metric changes, and updating agent state.
4. **Custom Workflow:** Inherit from `core.workflow.Workflow`. Implement `_setup_transitions` using `status_machine.add_state()` and `status_machine.add_transition()` with appropriate triggers (`StateTrigger`, `TimeTrigger`, `EventTrigger`).
5. **Custom DataProvider:** Inherit from `core.dataprovider.DataProvider`. Implement `load_data()` to retrieve data from external sources and `start_event_triggering()` to create SimPy processes that trigger events based on the loaded data at appropriate simulation times.

Using AirFogSim: A Collaborative Logistics Example

The following example demonstrates AirFogSim's capabilities in modeling collaborative logistics operations, a key application domain for LAVFC systems:

1. **Initialize Environment:** Create the simulation environment with appropriate visualization interval.

```
env = Environment(visual_interval=100)
```

2. **Create Delivery Stations:** Define multiple delivery stations with specific properties. The stations periodically generate payloads workflows and select drones for delivery.

```
station1 = DeliveryStation(
    env, "center_station",
    properties={
        'position': [0, 0, 0],
        'storage_capacity': 200,
        'service_radius': 100.0,
        'payload_generation_model': {...}
    }
)
```

3. **Create Delivery Drones:** Instantiate multiple drone agents with varying properties and components.

```
drone = DeliveryDroneAgent(
    env, "delivery_drone_1",
    properties={
        'position': [0, 0, 10],
        'battery_level': 90.0,
        'max_payload_weight': 5.0
    }
)
# Add components
```

```
drone.add_component(MoveToComponent(...)).
    add_component(LogisticsComponent(...))
env.register_agent(drone)
```

4. **Run Simulation and Analyze Results:** Execute the simulation and examine outcomes.

```
env.run(until=3600)
```

Visualization and Analysis

AirFogSim also includes a web-based visualization system using FastAPI and React for real-time monitoring. It provides:

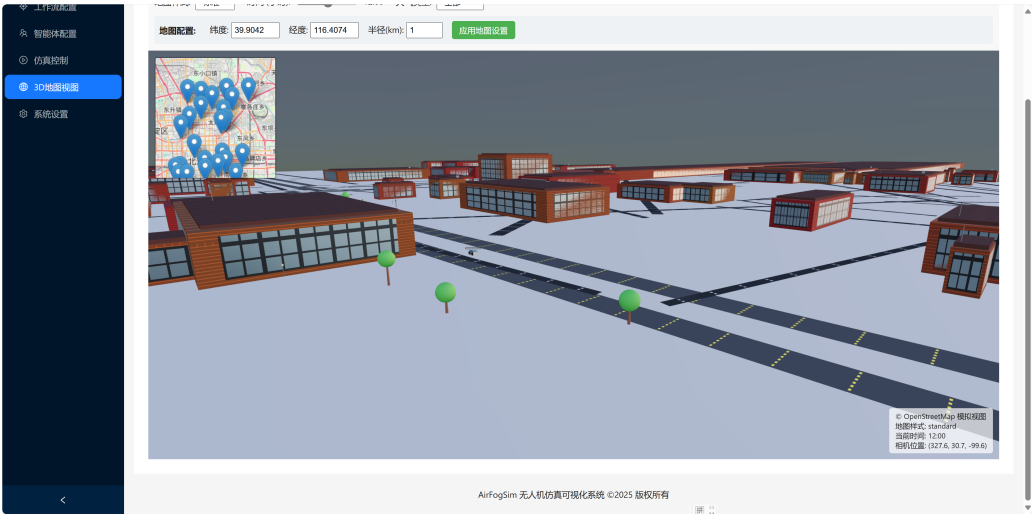


Figure 2: 3D visualization of UAVs in an urban environment with integrated traffic simulation.

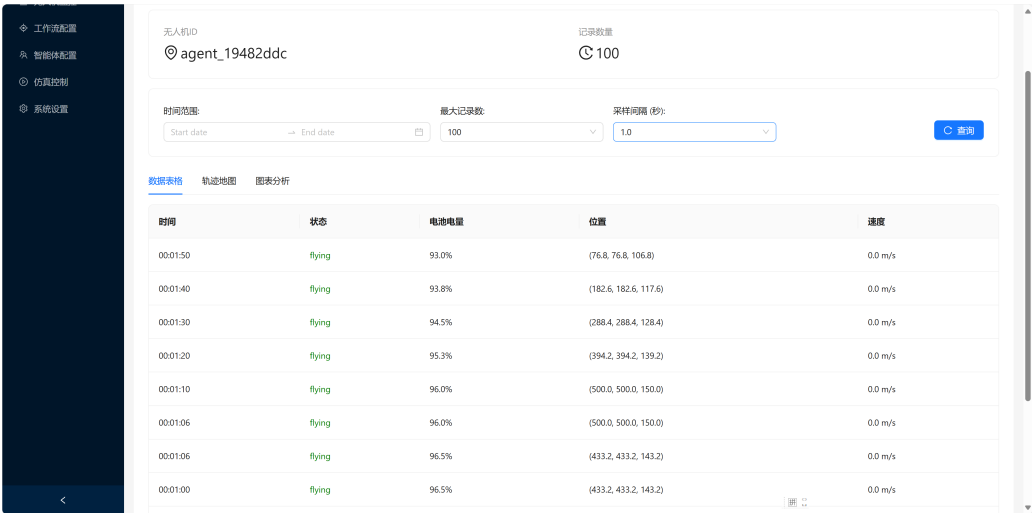


Figure 3: AirFogSim's monitoring interface showing UAV positions, states, and trajectories.

- 1. **3D Map View:** Shows agent positions, trajectories, resource locations.
- 2. **Dashboard:** Displays key performance indicators and system events.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant 62271351 and 62201390.

References

- Akgül, Ö. U., Mao, W., Cho, B., & Xiao, Y. (2023). VFogSim: A data-driven platform for simulating vehicular Fog computing environment. *IEEE Systems Journal*, 17(3), 5002–5013. <https://doi.org/10.1109/JSYST.2023.3286329>
- Gupta, H., Vahid Dastjerdi, A., Ghosh, S. K., & Buyya, R. (2017). iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Software: Practice and Experience*, 47(9), 1275–1296. <https://doi.org/10.1002/spe.2509>
- Hu, J., Chen, C., Cai, L., Khosravi, M. R., Pei, Q., & Wan, S. (2021). UAV-assisted vehicular Edge computing for the 6G Internet of Vehicles: Architecture, intelligence, and challenges. *IEEE Communications Standards Magazine*, 5(2), 12–18. <https://doi.org/10.1109/MCOMSTD.001.2000017>
- Huang, H., Su, J., & Wang, F.-Y. (2024). The potential of low-altitude airspace: The future of urban air transportation. *IEEE Transactions on Intelligent Vehicles*, 9(8), 5250–5254. <https://doi.org/10.1109/TIV.2024.3483889>
- Kong, F., Liu, X., Tang, B., Lin, J., Ren, Y., Cai, Y., Zhu, F., Chen, N., & Zhang, F. (2023). MARSIM: A light-weight point-realistic simulator for LiDAR-based UAVs. *IEEE Robotics and Automation Letters*, 8(5), 2954–2961. <https://doi.org/10.1109/LRA.2023.3264163>
- Krajzewicz, D., Erdmann, J., Behrisch, M., & Bieker-Walz, L. (2012). Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 5(3&4), 128–138. https://sumo.dlr.de/pdf/sysmea_v5_n34_2012_4.pdf
- Mahmud, R., Srirama, S. N., Ramamohanarao, K., & Buyya, R. (2022). iFogSim2: An extended iFogSim simulator for mobility, clustering, and microservice management in Edge and Fog computing environments. *Journal of Systems and Software*, 190, 111351. <https://doi.org/10.1016/j.jss.2022.111351>
- Mohebbi, S., & Murali, P. S. (2022). Spatial agent-based simulation of connected and autonomous vehicles to assess impacts on traffic conditions. *2022 Winter Simulation Conference (WSC)*, 2487–2498. <https://doi.org/10.1109/WSC57314.2022.10015452>
- Qayyum, T., Malik, A. W., Khattak, M. A. K., Khalid, O., & Khan, S. U. (2018). FogNet-Sim++: A toolkit for modeling and simulation of distributed Fog environment. *IEEE Access*, 6, 63570–63583. <https://doi.org/10.1109/access.2018.2877696>
- Sommer, C., Eckhoff, D., Brummer, A., Buse, D. S., Hagenauer, F., Joerer, S., & Segata, M. (2019). Veins: The open source vehicular network simulation framework. In A. Virdis & M. Kirsche (Eds.), *Recent advances in network simulation: The OMNeT++ environment and its ecosystem* (pp. 215–252). Springer International Publishing. https://doi.org/10.1007/978-3-030-12842-5_6
- Sonmez, C., Ozgovde, A., & Ersoy, C. (2017). EdgeCloudSim: An environment for performance evaluation of Edge computing systems. *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*, 39–44. <https://doi.org/10.1109/FMEC.2017.7946405>
- Wei, Z., Mao, J., Li, B., & Zhang, R. (2024). Privacy-preserving hierarchical reinforcement learning framework for task offloading in low-altitude vehicular Fog computing. *IEEE Open*

Journal of the Communications Society, 1–1. <https://doi.org/10.1109/OJCOMS.2024.3457023>