

# PiperABM: A Python Library for Resilience-Based Agent Modeling

Aslan Noorghasemi<sup>1</sup> and Christopher McComb<sup>1</sup>

<sup>1</sup> Department of Mechanical Engineering, Carnegie Mellon University, USA  Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

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

Editor: [Open Journals](#)

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

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

## Summary

PiperABM is an open-source Python library for building resilience-focused agent-based simulations on coupled infrastructure–society networks. Infrastructure (roads, homes, markets) is represented as a spatial graph that can progressively degrade; travel cost grows via an International Roughness Index (IRI)-inspired factor that inflates effective edge length over time, capturing both human wear and environmental stress. Agents decide with a lightweight Observe–Orient–Decide–Act loop and can be extended with custom policies, enabling rapid experimentation with behavioral assumptions. The library ingests real geospatial inputs (latitude/longitude from satellite or map data), projects them with a Mercator projection to a Cartesian grid, and instantiates city layouts directly in the model. Built-in utilities report resource accessibility and traveled distance, and the evolving system can be visualized via integrated animation rendering. PiperABM is a pure-Python, framework-agnostic package that interoperates with the scientific Python ecosystem (e.g., NumPy and Matplotlib), making it easy to embed in scripts and services.

## Statement of Need

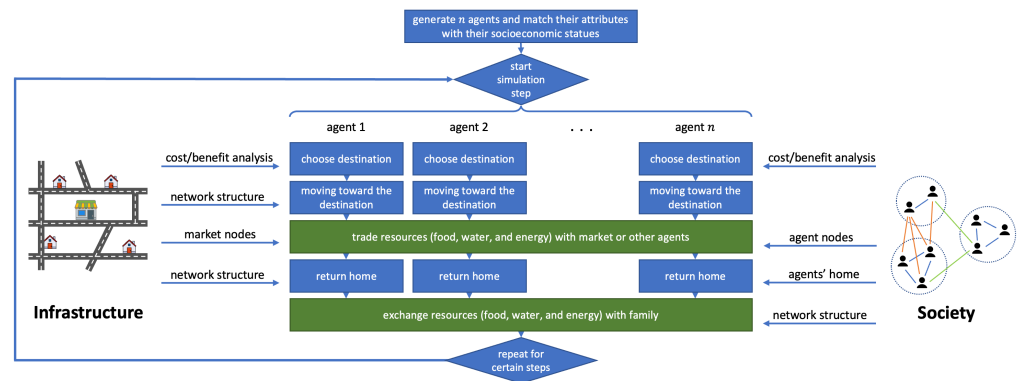
Infrastructure resilience is a critical concern for urban planners and researchers seeking to understand how disruptions (e.g., natural hazards) affect community access to essential services.

## How it works

The `piperabm` framework couples two dynamically interacting networks:

1. **A spatial infrastructure network** that carries physical flows, like roads for mobility and a simplified Food–Energy–Water (FEW) supply chain for basic needs.
2. **A social network of autonomous agents** whose daily decisions, movements, and exchanges both depend on and reshape the physical system.

The elements of these networks affect each other during each step of the simulation run, reflecting the intertwined nature of the dynamics. This simulation emulates the day-to-day life of the community, capturing how individuals interact with both infrastructure and each other as they pursue essential activities and respond to ongoing changes in their environment.



**Figure 1:** The computational model emulates the relation between the elements of infrastructure and social networks.

### Infrastructure

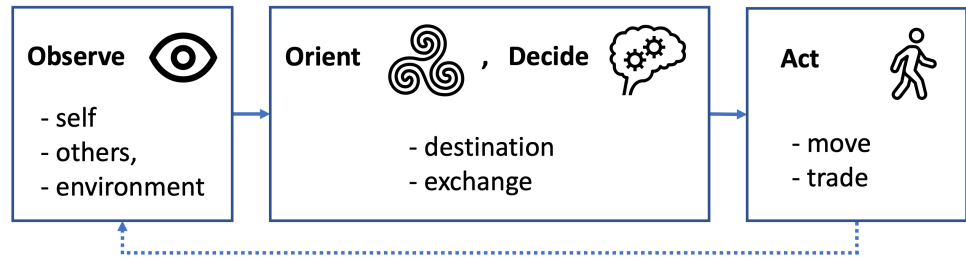
The modeling workflow begins by constructing a representation of the city's civil infrastructure system, hereafter referred to simply as the *infrastructure*. In the current work this term encompasses the physical assets that underpin service domains most critical to day-to-day community resilience, namely transportation and the Food–Energy–Water (FEW) supply chain. We abstract these assets into a spatial network comprising three classes of location nodes:

- **Junctions:** A placeholder in space with physical coordinates as their only attributes, such as where two streets intersect.
- **Homes:** Representing agents' residences, home nodes are where agents belong and reside within the community. These home nodes represent residential locations where agents reside, rest, and engage in family activities.
- **Markets:** These nodes represent local supermarkets that act as central hubs for the community's grocery shopping needs. The market nodes represent a generalized resource-influx point where essential food and goods enter the system, whether via imported supplies or local subsistence activities

{ I want to say it is possible to load city maps using satellite images and it has capability of converting lat long using mercator. }

### Society

The agents' decision-making processes in our model are governed by the OODA loop (Johnson, 2023). This framework, which stands for Observe, Orient, Decide, and Act, is particularly effective in modeling the complex cognitive behaviors of humans, as it encompasses a broad range of cognitive activities (Brehmer, 2005).



**Figure 2:** The agents' decision-making processes are modeled by OODA Loop which stands for Observe, Orient, Decide, and Act.

55 User can customize it by supplying their own `decision_making.py` modules.

## 56 Results

57 The result of simulation, for the matter of storage efficiency, is saved using data differencing  
58 (Noorghasemi & McComb, 2025). The transactions between agents are also saved separately.  
59 It is also possible to generate results below using the integrated functionalities.

## 60 Animation

61 When the time step is sufficiently coarse, PiperABM can export an MP4 animation of the  
62 evolving infrastructure graph and agent trajectories. These visualizations provide valuable *face*  
63 *validity*, a quick qualitative check that agents move sensibly, disruptions propagate plausibly,  
64 and global patterns match expectations.

## 65 Accessibility

66 Each agent's accessibility to resources is assessed at every time step to monitor their well-being  
67 and ability to meet their needs. The term accessibility  $A_{i,t,r}$  for agent  $i$  at time  $t$  for resource  
68  $r$  is computed as:

$$A_{i,t,r} = \frac{R_{i,t}}{R_i^{\max}}$$

69 where  $R_{i,t}$  is the amount of resource  $r$  that agent  $i$  possesses at time  $t$ , and  $R_i^{\max}$  is the  
70 maximum capacity of resource  $r$  that agent  $i$  can have. A value of 1 indicates full accessibility.

71 To aggregate across the  $R$  different resources for each agent, we use the geometric mean:

$$A_{i,t} = \left( \prod_{r=1}^R A_{i,t,r} \right)^{\frac{1}{R}}$$

72 This ensures that low accessibility in any single resource strongly impacts the overall score. If  
73 any  $A_{i,t,r} = 0$ , then  $A_{i,t} = 0$  and the agent is considered dead.

74 Across all  $N$  agents at each time step, the community's average accessibility is:

$$A_t = \frac{1}{N} \sum_{i=1}^N A_{i,t}$$

75 Finally, a time-weighted overall accessibility over the simulation duration  $T$  is

$$A = \frac{\int_0^T A_t dt}{\int_0^T A_{\max} dt}$$

76 where  $A_{\max} = 1$  is the maximum possible accessibility.

## 77 Travel Distance

78 In the context of agent-based modeling, *traveled distance* serves as a metric for assessing the  
79 efficiency and functionality of transportation networks within a simulated environment. This  
80 measurement tracks the cumulative distance agents must traverse between various points,  
81 e.g. from home to market.

82 When this measurement yields a low value, it indicates that the system is operating with  
83 high efficiency, allowing agents to traverse shorter distances between points to satisfy their  
84 needs. Alternatively, it could signal that various barriers, constraints, or issues are impeding  
85 agents' access to essential network nodes, thus limiting their ability to move freely within  
86 the system and reach their goals. This dual interpretation helps in diagnosing the underlying  
87 causes of system performance, guiding targeted improvements in urban planning and resource  
88 distribution.

## 89 Comparison to Existing Tools

90 PiperABM's strength lies in its opinionated support for resilience metrics, built-in animation  
91 utilities, and its minimal barrier for user-defined agent policies. Unlike Mesa or NetLogo, which  
92 require extensive boilerplate or domain-specific scripting, PiperABM users can implement  
93 new decision-making modules by inheriting from a common superclass. Compared to Repast,  
94 PiperABM remains lightweight and fully Pythonic, benefiting from the broad data science  
95 ecosystem without Java dependencies.

## 96 Acknowledgements

97 This work was supported by the U.S. National Science Foundation (Grant RISE-1927718).

## 98 References

- 99 Brehmer, B. (2005). The dynamic OODA loop: Amalgamating boyd's OODA loop and the  
100 cybernetic approach to command and control. *Proceedings of the 10th International*  
101 *Command and Control Research Technology Symposium*, 365–368.
- 102 Johnson, J. (2023). Automating the OODA loop in the age of intelligent machines: Reaffirming  
103 the role of humans in command-and-control decision-making in the digital age. *Defence*  
104 *Studies*, 23(1), 43–67. <https://doi.org/10.1080/14702436.2022.2102486>
- 105 Noorghasemi, A., & McComb, C. (2025). KeepDelta: A Python Library for Human-Readable  
106 Data Differencing. *Journal of Open Source Software*, 10(110), 8075. [https://doi.org/10.](https://doi.org/10.21105/joss.08075)  
107 [21105/joss.08075](https://doi.org/10.21105/joss.08075)