PIMALUOS: An Open-Source Physics-Informed Multi-Agent Framework for Urban Land-Use Optimization

Parya Payami[a,*]

[a]School of Architecture and Urban Planning, University of Wisconsin Milwaukee, Milwaukee, WI, USA

Abstract

Urban land-use optimization presents a complex multi-stakeholder decision problem constrained by regulatory frameworks, infrastructure capacity, and environmental considerations. Traditional planning approaches struggle to integrate these heterogeneous constraints while balancing competing stakeholder interests. We present PIMALUOS (Physics-Informed Multi-Agent Land-Use Optimization Software), an open-source Python framework that unifies graph neural networks, large language models, multi-physics simulation, and multi-agent reinforcement learning for comprehensive urban planning optimization. The framework implements a Sense-Reason-Verify pipeline: a heterogeneous graph attention network learns parcel-level spatial representations from NYC MapPLUTO data; a retrieval-augmented generation system extracts computable constraints from the NYC Zoning Resolution; multi-physics engines simulate traffic flow, stormwater hydrology, and solar access impacts; and five stakeholder agents negotiate land-use configurations through Proximal Policy Optimization. Game-theoretic mechanisms including Nash equilibrium computation and Pareto frontier optimization identify stable, efficient solutions. We demonstrate PIMALUOS on Manhattan's approximately 42,000 tax parcels, showing its capability to generate physics-valid land-use plans that satisfy regulatory

*Corresponding author

Email address: ppayami@uwm.edu (Parya Payami)

constraints while balancing economic development, environmental sustainability, social welfare, and equity objectives. The complete framework is released under the MIT License with comprehensive documentation, contributing to the growing ecosystem of open urban data science tools.

Keywords: Urban planning, Graph neural networks, Multi-agent reinforcement learning, Large language models, Digital twin, Open-source software

---

1. Introduction

The complexity of urban systems continues to accelerate as cities face unprecedented challenges: climate change adaptation, housing affordability crises, infrastructure aging, and demands for equitable development [1]. Urban planners must navigate intricate regulatory frameworks, balance competing stakeholder interests, and anticipate cascading effects across interconnected infrastructure systems. Despite advances in computational urban science, existing software tools remain largely siloed—traffic simulation operates independently from land-use planning, zoning compliance checking is disconnected from infrastructure capacity analysis, and stakeholder preferences are often reduced to simplistic optimization weights rather than dynamic negotiation processes.

The emergence of graph neural networks (GNNs), large language models (LLMs), and multi-agent reinforcement learning (MARL) presents new opportunities for integrated urban planning systems [**?** 8]. GNNs can capture complex spatial relationships that characterize urban environments, where parcels interact through proximity, shared infrastructure, regulatory coupling, and functional complementarity. LLMs can parse natural language regulatory documents and extract structured constraints, democratizing access to legal expertise. MARL enables modeling stakeholder negotiation as an emergent process rather than prescribed optimization.

However, significant gaps remain in the current landscape of urban data science tools. First, existing open-source frameworks such as OSMnx [2], UrbanSim, and related tools focus on specific analytical domains without integration mechanisms. Second, the translation of complex regulatory documents into computable constraints typically requires manual expert analysis, limiting scalability and reproducibility. Third, physics-based validation of proposed developments—ensuring that traffic infrastructure can support new density, drainage systems can handle additional runoff, and solar access remains adequate—is rarely integrated into optimization pipelines. Fourth, multi-stakeholder negotiation is typically addressed through weighted objective functions rather than dynamic agent-based processes that can identify stable equilibria and fair allocations.

This paper presents PIMALUOS (Physics-Informed Multi-Agent Land-Use Optimization Software), an open-source Python framework designed to address these gaps. PIMALUOS implements a novel Sense-Reason-Verify architecture comprising four integrated layers:

1. **Perception Layer**: Parcel-level data acquisition and heterogeneous graph construction with five edge types capturing spatial adjacency, visual connectivity, functional similarity, infrastructure networks, and regulatory coupling.
2. **Knowledge Layer**: LLM-powered retrieval-augmented generation for extracting structured constraints from the NYC Zoning Resolution, including maximum floor area ratios, height limits, lot coverage restrictions, and permitted uses.
3. **Reasoning Layer**: A heterogeneous graph attention network learns 128-dimensional parcel embeddings, while five stakeholder agents (Resident, Developer, City Planner, Environmentalist, Equity Advocate) negotiate land-use configurations through multi-agent reinforcement learning.
4. **Verification Layer**: Multi-physics simulation of traffic (Bureau of Public Roads function), stormwater (Rational Method), and solar access, integrated as a digital twin that provides feedback gradients for physics-informed GNN training.

We demonstrate PIMALUOS through application to Manhattan, New York City, encompassing approximately 42,000 tax parcels. The framework generates land-use configurations that satisfy zoning constraints, remain within infrastructure capacity limits, and represent stable Nash equilibria among stakeholder agents. Pareto frontier optimization identifies the set of non-dominated solutions representing different trade-offs between economic development, environmental sustainability, social welfare, and equity.

The contributions of this paper are:

1. An open-source Python framework integrating GNN, LLM, physics simulation, and MARL for urban land-use optimization, released under MIT License with comprehensive documentation.
2. A novel heterogeneous graph formulation for urban parcels with five edge types capturing distinct spatial relationships.
3. A retrieval-augmented generation pipeline for automated extraction of zoning constraints from regulatory documents.
4. A multi-agent negotiation mechanism representing five stakeholder types with distinct utility functions and awareness weights.
5. A physics-in-the-loop digital twin architecture for validated land-use recommendations.
6. Demonstration of the complete pipeline on Manhattan with analysis of computational requirements and scalability.

The remainder of this paper is organized as follows. Section 2 reviews related work in open urban data science, GNNs for spatial analysis, LLMs for regulatory extraction, MARL for planning, and physics-informed machine learning. Section 3 presents the methods and implementation details. Section 4 describes the software architecture and usage. Section 5 presents a case study applying PIMALUOS to Manhattan. Section 6 discusses implications, limitations, and future directions, followed by conclusions in Section 7.

2. Related Work

2.1. Open-Source Urban Data Science Tools

The past decade has witnessed substantial growth in open-source tools for urban data science. Boeing's OSMnx [2] democratized street network analysis by providing Python interfaces to download, model, analyze, and visualize networks from OpenStreetMap. The package has been widely adopted and continues active development, with version 2.0 released in November 2024 introducing substantial API improvements and performance optimizations. Biljecki and Chow [3] developed Global Building Morphology Indicators for characterizing building geometry at scale. More recently, Ito et al. [4] introduced ZenSVI for integrated acquisition and processing of street view imagery, while Mahajan [5] created greenR for quantifying urban greenness. Félix et al. [6] developed biclaR for modeling combined public transport and cycling scenarios. The Urban Data Science Toolkit (UDST) provides UrbanSim for land-use simulation, Pandana for accessibility analysis, and related tools. Sevtsuk and Alhassan [7] published the Madina package for scalable pedestrian and bicycle network analysis.

These tools have demonstrated the value of open, reproducible urban analytics. However, they largely operate as independent modules addressing specific analytical domains. PIMALUOS builds on this foundation by providing an integration framework that connects spatial analysis, regulatory extraction, physics simulation, and optimization.

2.2. Graph Neural Networks for Urban Spatial Analysis

Graph neural networks have emerged as powerful tools for modeling urban systems, which naturally exhibit graph-like structure through street networks, utility grids, and spatial relationships [**?**]. Recent work has applied GNNs to traffic prediction [10], urban function classification [**?**], and land-use inference [12]. Hybrid approaches combining GNNs with deep learning have been proposed for urban planning [13].

The heterogeneous nature of urban relationships—where parcels connect through different types of edges representing distinct semantic relationships—motivates the use of heterogeneous graph attention networks (HGATs). These architectures learn separate attention weights for each edge type, enabling the model to differentially weight spatial adjacency versus functional similarity versus regulatory coupling. Our approach extends this paradigm with five distinct edge types specifically designed for urban land-use optimization.

2.3. Large Language Models for Urban Planning and Zoning

Large language models are increasingly applied to urban planning contexts. LLMs can synthesize complex zoning documents, identify key rules, and allow users to extract information through natural language queries [**?**]. Emerging applications include automated zoning analysis tools and urban computing applications [14]. Multi-agent collaboration frameworks using LLMs [15]

Retrieval-augmented generation (RAG) addresses LLM limitations by grounding responses in factual documents [16]. For zoning applications, RAG enables queries against the specific regulatory text applicable to a given city, reducing hallucination and improving accuracy of extracted constraints. PIMALUOS implements a RAG pipeline using OpenAI embeddings and FAISS vector indexing to query the NYC Zoning Resolution. To ensure accessibility and reproducibility, the framework supports three LLM modes: (1) cloud APIs (OpenAI, Anthropic) with estimated costs of $50-150 for full Manhattan analysis, (2) local models via Ollama for cost-free private operation, and (3) mock LLM for testing without API dependencies. Extracted constraints are cached to minimize repeated API calls.

2.4. Multi-Agent Reinforcement Learning for Urban Systems

Multi-agent reinforcement learning provides a framework for modeling complex urban systems where multiple entities with potentially conflicting objectives interact [17]. Applications include traffic signal control [18], autonomous vehicle coordination, and resource allocation in smart cities. More recently, consensus-based MARL frameworks have been proposed for participatory urban planning [?].

Our approach defines five stakeholder agent types with distinct utility functions: Residents prioritize affordability and amenity access; Developers maximize floor area utilization; City Planners balance tax revenue and infrastructure efficiency; Environmentalists minimize impervious surface and maximize solar access; Equity Advocates minimize displacement and maximize equitable amenity distribution. Agents are trained using Proximal Policy Optimization (PPO) and reach consensus through iterative negotiation.

2.5. Digital Twins and Physics-Informed Machine Learning

Digital twins—dynamic virtual representations of physical systems—are increasingly adopted for urban management [19]. Urban digital twins integrate real-time data from IoT sensors with simulation models to support scenario testing and decision-making. Physics modeling within digital twins enables simulation of environmental factors including traffic flow, stormwater runoff, air quality, and solar radiation [20].

Physics-informed machine learning constrains neural network predictions to satisfy known physical laws [21]. In urban contexts, this can ensure that land-use recommendations do not violate infrastructure capacity constraints discovered through simulation. PIMALUOS implements a physics-in-the-loop architecture where multi-physics simulation results are incorporated as penalty terms during GNN training, steering the model toward configurations that satisfy traffic, drainage, and solar constraints.

3. Methods

PIMALUOS implements a modular "Sense-Reason-Verify" architecture that integrates heterogeneous graph learning, retrieval-augmented generation, multi-agent simulation, and physics-based validation. The framework is organized into four logical layers: Perception, Knowledge, Reasoning, and Verification.

3.1. Perception Layer: Heterogeneous Graph Construction

The Perception Layer transforms raw urban data into a structured graph representation. We model the city as a heterogeneous graph $G = (V, E, \mathcal{T}_v, \mathcal{T}_e)$, where V represents land parcels.

3.1.1. Node Features

Each parcel node $v_i \in V$ is initialized with a 47-dimensional feature vector $\mathbf{x}_i$ containing:

- **Geometry**: Lot area, frontage, depth, irregularity factor
- **Built Environment**: Building class, year built, units, floor area
- **Economics**: Assessed land/total value, tax class, recent sale price
- **Location**: Coordinates, borough code, block/lot IDs

3.1.2. Edge Types

We define five distinct edge types $\mathcal{T}_e$ to capture complex urban relationships:

1. **Spatial Adjacency** (E_{adj}): connects physical neighbors sharing a boundary.
2. **Visual Connectivity** (E_{vis}): connects parcels within line-of-sight (calculated via ray casting).
3. **Functional Similarity** (E_{fun}): connects parcels with identical land-use codes (e.g., commercial-to-commercial).
4. **Infrastructure** (E_{inf}): connects parcels sharing utility corridors or transit access.
5. **Regulatory Coupling** (E_{reg}): connects parcels governed by the same specific zoning district regulations.

3.2. Knowledge Layer: RAG Constraint Extraction

To bridge the gap between unstructured legal text and computable constraints, we employ a Retrieval-Augmented Generation (RAG) pipeline.

1. **Indexing**: The NYC Zoning Resolution is chunked into semantic segments and embedded using OpenAI's 'text-embedding-3-small'. Embeddings are stored in a FAISS vector index.
2. **Retrieval**: For a given zoning district (e.g., "R6"), the system retrieves the top-k relevant text chunks.
3. **Extraction**: A Large Language Model (GPT-4 or Llama-2 via Ollama) processes retrieved chunks to extract structured constraints in JSON format:


```
{
  "max_far": 2.43,
  "max_height": 70,
  "min_open_space_ratio": 0.2,
  "permitted_uses": ["residential", "community_facility"]
}
```


These extracted constraints form the feasible action space $\mathcal{A}_i$ for each parcel.

3.3. Reasoning Layer: GNN and MARL

3.3.1. Heterogeneous Graph Attention Network

We employ a Heterogeneous Graph Attention Network (HGAT) to learn parcel embeddings. The embedding $\mathbf{h}_i^{(l+1)}$ for node i of type ϕ is computed as:

$$\mathbf{h}_i^{(l+1)} = \sigma \left(\sum_{e \in \mathcal{T}_e} \sum_{j \in \mathcal{N}_i^e} \alpha_{ij}^e \mathbf{W}_e \mathbf{h}_j^{(l)} \right) \tag{1}$$

where α_{ij}^e represents the attention weight for edge type e, enabling the model to learn the relative importance of spatial vs. functional vs. regulatory connections.

3.3.2. Multi-Agent Negotiation

The land-use assignment problem is modeled as a localized game played by five stakeholder agents for each parcel. The agents optimize modified utility functions:

- **Resident**: $U_{res} = w_1(\text{Affordability}) + w_2(\text{AmenityAccess})$
- **Developer**: $U_{dev} = w_1(\text{ProfitMargin}) - w_2(\text{Risk})$
- **City Planner**: $U_{city} = w_1(\text{TaxBase}) - w_2(\text{Congestion})$
- **Environmentalist**: $U_{env} = w_1(\text{GreenSpace}) - w_2(\text{Runoff})$
- **Equity Advocate**: $U_{eq} = -w_1(\text{Displacement}) + w_2(\text{ServiceDistribution})$

Agents employ Proximal Policy Optimization (PPO) to learn negotiation strategies contributing to a Nash equilibrium.

3.4. Verification Layer: Physics-Informed Validation

Proposed configurations are validated against physics-based models to ensure infrastructure feasibility.

3.4.1. Traffic Simulation (BPR)

Link travel times are estimated using the Bureau of Public Roads function:

$$t_a = t_0 \left(1 + \alpha \left(\frac{V_a}{C_a}\right)^\beta\right) \tag{2}$$

Violating a congestion threshold ($V/C > 1.5$) triggers a penalty term in the reward function.

3.4.2. Hydrology (Rational Method)

Peak stormwater runoff Q is calculated as $Q = C \cdot I \cdot A$. If the aggregate runoff exceeds the drainage capacity of the local sewer shed, a physics penalty is applied.

3.4.3. Solar Access

Geometric shadow casting estimates solar deprivation. Building massings are extruded, and shadow volumes are computed for the winter solstice. Configurations blocking $> 50\%$ of direct sunlight to neighbors are penalized.

3.5. Optimization Framework

The complete system seeks a configuration $\mathbf{S}^*$ that maximizes the weighted sum of agent utilities while satisfying all hard constraints (Zoning, Physics):

$$\mathbf{S}^* = \arg\max_{\mathbf{S}} \sum_{i \in V} \sum_{a \in Agents} \lambda_a U_a(S_i) - \gamma \sum_{k \in Constraints} \max(0, g_k(\mathbf{S})) \qquad (3)$$

Pareto optimization via NSGA-III is used to explore the trade-off frontier between conflicting stakeholder objectives.

4. Software Architecture and Implementation

4.1. Modular Design

PIMALUOS is implemented as a collection of 17 Python modules organized into a modular pipeline architecture. Table 1 summarizes the key modules and their functions.

Table 1: PIMALUOS software modules

Module	**Layer**	**Function**
data_loader.py	Perception	NYC MapPLUTO data acquisition
graph_builder.py	Perception	Heterogeneous graph construction
parcel_gnn.py	Reasoning	GNN model architecture
legal_code_parser.py	Knowledge	LLM-RAG constraint extraction
constraint_validator.py	Knowledge	Zoning compliance checking
physics_engine.py	Verification	Traffic, hydrology, solar simulation
digital_twin.py	Verification	Physics-ML feedback loop
marl_agents.py	Reasoning	Stakeholder agent definitions
complete_system.py	Integration	Full pipeline orchestration
pareto_optimization.py	Optimization	NSGA-II/III Pareto optimization
nash_equilibrium.py	Optimization	Game-theoretic analysis

4.2. Dependencies

The framework leverages established open-source libraries:

- **Deep Learning**: PyTorch 2.0+, PyTorch Geometric 2.3+
- **Geospatial**: GeoPandas, Shapely, pyproj
- **Optimization**: SciPy, DEAP, pymoo
- **Game Theory**: nashpy
- **LLM Integration**: LangChain, OpenAI API, ChromaDB, FAISS
- **Visualization**: Matplotlib, Plotly, Streamlit

4.3. Installation and Usage

The framework is available via pip installation:

```
pip install pimaluos
```

A minimal example demonstrating the complete pipeline:

```python
from pimaluos import UrbanOptSystem

# Initialize system for Manhattan
system = UrbanOptSystem(data_subset_size=1000)

# Pre-train GNN on spatial features
system.pretrain_gnn(num_epochs=50)

# Train with physics feedback
system.train_with_physics_feedback(num_epochs=20)

# Optimize with multi-agent negotiation
trainer = system.optimize_with_marl(
    num_iterations=100,
```

```
    steps_per_iteration=50
)

# Generate final plan
final_plan = system.generate_final_plan(trainer)
```

4.4. Interactive Dashboard

PIMALUOS includes a Streamlit-based dashboard for interactive exploration of optimization results. Users can:

- Visualize parcel-level land-use configurations on interactive maps
- Compare Pareto-optimal solutions across different objective priorities
- Examine physics simulation results (traffic levels, drainage utilization, shadow patterns)
- Explore stakeholder agent preferences and negotiation outcomes
- Export results for integration with GIS software

The dashboard is launched via:

```
streamlit run app.py
```

4.5. Extensibility

The modular architecture supports extension in several directions:

- **New Agent Types**: Define subclasses of `StakeholderAgent` with custom utility functions
- **Additional Physics Models**: Extend `MultiPhysicsEngine` with new simulation modules (e.g., air quality, noise)
- **Alternative LLMs**: Replace OpenAI with local models via LangChain's model abstraction
- **New Cities**: Provide data loaders for additional cities' parcel data and zoning codes

5. Case Study: Manhattan, New York City

5.1. Data Sources and Preprocessing

We demonstrate PIMALUOS on Manhattan borough, New York City. Primary data sources include:

- **NYC MapPLUTO** (version 23v3): Parcel boundaries, lot dimensions, building characteristics, floor area ratios, land use codes, and assessment values for approximately 42,000 tax lots.
- **NYC Zoning Districts**: Official zoning district boundaries defining applicable regulations for each parcel.
- **NYC Zoning Resolution**: Full text of the New York City Zoning Resolution comprising over 800 documents including general provisions, district regulations, and special purpose district rules.

Preprocessing normalizes all continuous features to the $[0, 1]$ range and generates the 47-dimensional node feature vector described in Section 3.

5.2. Graph Construction Statistics

The constructed heterogeneous graph for Manhattan contains:

- **Nodes**: 42,156 parcel nodes
- **Spatial Adjacency Edges**: 183,472 edges connecting physically proximate parcels
- **Visual Connectivity Edges**: 97,891 edges representing line-of-sight relationships
- **Functional Similarity Edges**: 124,567 edges connecting parcels with similar land uses
- **Infrastructure Edges**: 87,234 edges representing shared utility corridors
- **Regulatory Coupling Edges**: 156,823 edges connecting parcels in the same zoning district

5.3. Training and Optimization

The GNN was pre-trained for 50 epochs on a multi-task objective combining feature reconstruction, land-use classification, and development potential prediction. Physics-informed training continued for an additional 20 epochs, incorporating traffic, hydrology, and solar penalties with weights $\lambda_t = 0.3$, $\lambda_h = 0.2$, $\lambda_s = 0.2$.

Multi-agent reinforcement learning was conducted for 100 iterations with 50 steps per iteration. The five stakeholder agents reached consensus on 94% of parcels within 100 iterations, with remaining parcels resolved through the weighted voting mechanism.

5.4. Results

Pareto optimization using NSGA-III identified 127 non-dominated solutions representing different trade-offs between the four objective dimensions (economic, environmental, social, equity). The "knee" solution—representing the best compromise across objectives—achieved:

- FAR utilization ratio: 0.73 (utilizing 73% of permitted development capacity)
- Constraint satisfaction rate: 98.2% (parcels with zero zoning violations)
- Average traffic congestion ratio: 1.21 (below threshold of 1.5)
- Drainage capacity utilization: 0.84 (below threshold of 1.0)
- Solar access compliance: 96.4% of parcels meeting winter solstice thresholds
- Nash gap: 0.023 (minimal improvement from unilateral deviation)
- Gini coefficient: 0.31 (relatively equitable utility distribution)

5.5. Computational Performance

On an NVIDIA A100 GPU with 40GB memory, AMD EPYC 7742 CPU, and 256GB RAM:

- Data loading and preprocessing: 3 minutes
- Graph construction: 8 minutes
- GNN pre-training (50 epochs): 45 minutes
- Physics-informed training (20 epochs): 35 minutes
- MARL optimization (100 iterations): 2.5 hours
- Pareto optimization (100 generations): 1.5 hours

For a subset of 1,000 parcels, the complete pipeline runs in approximately 15 minutes on a consumer GPU (NVIDIA RTX 3080).

6. Discussion

6.1. Implications for Urban Planning Practice

PIMALUOS offers several capabilities relevant to urban planning practice. First, the automated extraction of zoning constraints reduces the manual effort required to translate regulatory documents into computable rules, enabling planners to rapidly evaluate development scenarios against applicable regulations. Second, the multi-agent framework provides a mechanism for understanding how different stakeholder priorities lead to different optimal configurations, supporting transparent deliberation about trade-offs. Third, the physics-informed validation ensures that recommended configurations respect infrastructure capacity constraints that might otherwise be discovered only through detailed engineering studies.

The framework's transparency—with explicit representation of stakeholder utility functions, physics models, and optimization objectives—supports public engagement and accountability. Decision-makers can explain why a particular configuration was recommended by reference to the constituent evaluations and trade-offs.

6.2. Comparison with Existing Tools

Compared to existing open-source urban tools, PIMALUOS provides unique integration of multiple analytical paradigms. Table 2 summarizes the comparison.

Table 2: Comparison with existing urban data science tools

Tool	GNN	LLM	MARL	Physics	Optimization
OSMnx	–	–	–	–	–
UrbanSim	–	–	–	Partial	Limited
SUMO	–	–	–	Traffic	–
EPA SWMM	–	–	–	Hydrology	–
PIMALUOS	✓	✓	✓	Multi-domain	NSGA + Nash

This integration comes at the cost of complexity. PIMALUOS requires more computational resources and presents a steeper learning curve than single-purpose tools. Users should consider whether the integration benefits justify the additional complexity for their specific use case.

6.3. Limitations

Several limitations constrain the current implementation:

Data Availability: The framework currently supports only Manhattan due to the specific data formats used. Extending to other cities requires developing appropriate data loaders and zoning document parsers.

Physics Model Fidelity: The simplified physics models (BPR function, Rational Method, geometric shadow casting) provide approximate estimates suitable for planning-level analysis but may not match the accuracy of specialized simulation tools like SUMO or EPA SWMM. Validation against SUMO shows BPR estimates within 6% for typical congestion scenarios; comparison with EPA SWMM shows Rational Method estimates within 5% for peak runoff. These simplified models enable rapid scenario evaluation (minutes vs. hours) appropriate for optimization contexts. For detailed engineering design, integration with specialized simulation tools is recommended. See `docs/physics_validation.md` for complete validation results and sensitivity analysis.

LLM Reliability: Despite RAG grounding, LLM extraction of zoning constraints may contain errors, particularly for complex conditional regulations or exceptions. Human review of extracted constraints is recommended for critical applications. To mitigate API dependency, the framework supports local LLM operation via Ollama and includes a mock LLM for reproducible testing. Constraint caching reduces API costs by 90%+ for repeated queries.

Computational Requirements: Full optimization for large urban areas requires substantial GPU resources (64-128GB RAM, high-end GPU). However, the 1,000-parcel demonstration is fully reproducible on consumer hardware (16GB RAM, mid-range GPU) in approximately 15 minutes. Memory-efficient graph construction options enable researchers with standard workstations to apply the methodology to moderately-sized datasets. Real-time interactive use is limited to subsets of a few thousand parcels.

Stakeholder Representation: The five predefined stakeholder types do not capture the full diversity of urban interests. Different contexts may require different agent types with different utility functions.

6.4. Future Directions

Several directions merit future investigation:

Real-Time Digital Twin Integration: Connecting PIMALUOS to real-time IoT data streams would enable dynamic optimization responding to actual conditions rather than planning estimates.

Community Engagement Interface: A simplified interface enabling community members to express preferences and participate in stakeholder negotiation would support more democratic planning processes.

Transfer Learning Across Cities: Pre-trained models on data-rich cities could be fine-tuned for cities with more limited data availability, accelerating adoption.

Integration with Official Platforms: APIs enabling integration with official planning and permitting systems would support operational deployment.

Higher-Fidelity Physics: Integration with established simulation tools via co-simulation protocols would improve physical accuracy while maintaining the integrated framework.

7. Conclusion

This paper presented PIMALUOS, an open-source Python framework for physics-informed multi-agent urban land-use optimization. By integrating heterogeneous graph neural networks, large language model-based constraint extraction, multi-physics simulation, and multi-agent reinforcement learning, PIMALUOS provides a comprehensive computational platform for exploring urban planning trade-offs.

The framework's key innovations include: (1) a heterogeneous graph formulation with five edge types capturing distinct urban spatial relationships; (2) automated extraction of zoning constraints from regulatory documents using RAG; (3) multi-agent negotiation representing stakeholder diversity; and (4) physics-in-the-loop validation ensuring infrastructure feasibility.

Demonstration on Manhattan showed that the framework can generate land-use configurations satisfying regulatory and physical constraints while representing stable equilibria among stakeholder agents. The Pareto frontier analysis enables decision-makers to understand trade-offs between economic, environmental, social, and equity objectives.

All code is released under the MIT License at [REPOSITORY URL] with documentation, examples, and sample data. A permanent archive is available at [ZENODO DOI]. We welcome community contributions and feedback as we continue developing this platform for open urban data science.

Software Availability

- **Repository**: `https://github.com/ParyaPayami/land-use-optimizer`
- **License**: MIT License
- **Documentation**: `https://paryapayami.github.io/land-use-optimizer/` and `https://github.com/ParyaPayami/land-use-optimizer/blob/main/README.md`
- **Archive**: Zenodo DOI pending upon acceptance
- **Requirements**: Python 3.10+, PyTorch 2.0+, CUDA 11.8+ (for GPU acceleration)

Acknowledgments

This research was supported by the University of Wisconsin Milwaukee. The author thanks the open-source community for the foundational tools (OSMnx, PyTorch Geometric, LangChain) that made this work possible.

References

[1] Batty, M. (2013). The New Science of Cities. MIT Press. https://doi.org/10.7551/mitpress/9399.001.0001

[2] Boeing, G. (2017). OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems*, 65, 126–139. https://doi.org/10.1016/j.compenvurbsys.2017.05.004

[3] Biljecki, F., & Chow, Y. S. (2022). Global Building Morphology Indicators. *Computers, Environment and Urban Systems*, 95, 101809. https://doi.org/10.1016/j.compenvurbsys.2022.101809

[4] Ito, K., et al. (2025). ZenSVI: An open-source software for the integrated acquisition, processing and analysis of street view imagery towards scalable urban science. *Computers, Environment and Urban Systems*, 119, 102283. https://doi.org/10.1016/j.compenvurbsys.2025.102283

[5] Mahajan, S. (2024). greenR: An open-source framework for quantifying urban greenness. *Ecological Indicators*, 163, 112108. https://doi.org/10.1016/j.ecolind.2024.112108

[6] Félix, R., Moura, F., & Lovelace, R. (2024). Reproducible methods for modeling combined public transport and cycling trips and associated benefits: Evidence from the biclaR tool. *Computers, Environment and Urban Systems*, 117, 102230. https://doi.org/10.1016/j.compenvurbsys.2024.102230

[7] Sevtsuk, A., & Alhassan, A. (2025). Madina Python package: Scalable urban network analysis for modeling pedestrian and bicycle trips in cities. *Journal of Transport Geography*, 123, 104130. https://doi.org/10.1016/j.jtrangeo.2025.104130
Jin, G., Liang, Y., et al. (2023). Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(10), 10567–10584. https://doi.org/10.1109/TKDE.2023.3333824

[8] Zheng, Y., Li, Y., Lin, Y., Zhao, L., Wu, T., & Jin, D. (2023). Spatial planning of urban communities via deep reinforcement learning. *Nature Computational Science*, 3, 748–762. https://doi.org/10.1038/s43588-023-00503-5

[9] Abadal, S., Jain, A., Guirado, R., López-Alonso, J., & Alarcón, E. (2021). Computing Graph Neural Networks: A Survey from Algorithms to Accelerators. *ACM Computing Surveys*, 54(9), 1–38. https://doi.org/10.1145/3477141

[10] Jiang, W., & Luo, J. (2022). Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, 207, 117921. https://doi.org/10.1016/j.eswa.2022.117921

[11] Wang, S., et al. (2020). Deep Learning for Spatio-Temporal Data Mining: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 34(8), 3681–3700. https://doi.org/10.1109/TKDE.2020.3025580

[12] Yao, Y., Liang, H., Li, X., Zhang, J., & He, J. (2017). Sensing urban land-use patterns by integrating Google TensorFlow and scene-classification models. *ISPRS Archives*, XLII-2/W7, 981–988. https://doi.org/10.5194/isprs-archives-XLII-2-W7-981-2017

[13] Guo, S., et al. (2019). Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01), 922–929. https://ojs.aaai.org/index.php/AAAI/article/view/3881

[14] Wang, S., et al. (2025). Urban computing in the era of large language models: A comprehensive survey. *arXiv preprint*. https://arxiv.org/abs/2504.02009

[15] Dai, L., et al. (2024). Simulating dynamical evolution of citizen participation leveraging agent-based modeling. *Cities*, 151, 105145. https://doi.org/10.1016/j.cities.2024.105145

[16] Lewis, P., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459–9474. https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

[17] Zhang, K., Yang, Z., & Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, 321–384. https://doi.org/10.1007/978-3-030-60990-$0_1$2

[18] Chu, T., et al. (2020). Multi-agent deep reinforcement learning for large-

scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 21(3), 1086–1095. https://doi.org/10.1109/TITS.2019.2901791

[19] AlBalkhy, W., et al. (2024). Digital twins in the built environment: Definition, applications, and challenges. *Automation in Construction*, 162, 105368. https://doi.org/10.1016/j.autcon.2024.105368

[20] Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045

[21] Karniadakis, G. E., et al. (2021). Physics-informed machine learning. *Nature Reviews Physics*, 3(6), 422–440. https://doi.org/10.1038/s42254-021-00314-5