



Title	Data-Driven Framework for Agent-Based Pedestrian Modeling & Simulation
Author(s)	Sanjjamts, Amartaivan
Citation	大阪大学, 2026, 博士論文
Version Type	VoR
URL	https://doi.org/10.18910/105496
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

Data-Driven Framework for Agent-Based Pedestrian Modeling & Simulation

Submitted to
Graduate School of Information Science and Technology
The University of Osaka

January 2026

Sanjjamts AMARTAIVAN

List of Publications

Journal Articles

- S. Amartaivan and H. Morita, “Efficient Calibration of Heterogeneous Agent-Based Pedestrian Model Using PSO With Deep Learning Surrogates,” *IEEE Access*, vol. 13, pp. 109185–109199, 2025. doi: [10.1109/ACCESS.2025.3582778](https://doi.org/10.1109/ACCESS.2025.3582778)
- Amartaivan Sanjjamts and Hiroshi Morita, “Preliminary Study on Space Utilization and Emergent Behaviors of Group vs. Single Pedestrians in Real-World Trajectories,” *arXiv preprint*, August 2025. arXiv: [2508.18939](https://arxiv.org/abs/2508.18939) [cs.CV]

Conference Presentations

- Amartaivan Sanjjamts, “Data-Driven Agent-Based Pedestrian Modeling and Simulation,” *Operations Research Society of Japan (ORSJ), Kansai – Young Researchers Symposium*, November 2024, Osaka, Japan.
- Amartaivan Sanjjamts, “Data-Driven Agent-Based Pedestrian Modeling & Simulation in Indoor Environments,” *Proceedings of the 2024 Winter Simulation Conference*, December 2024, Florida, USA.
- Amartaivan Sanjjamts, Hiroshi Morita, “Impact of Scheduling Methods on Agent-Based Evacuation Dynamics,” *International Symposium on Scheduling 2025*, June 2025, Tokyo, Japan.

Abstract

Understanding pedestrian dynamics is fundamental to the development of safe, efficient, and inclusive urban environments. With continued urbanization, cities face increasing challenges in managing high-density pedestrian flows within limited spatial infrastructures. The effective design of walkways, intersections, and public spaces, therefore, requires accurate modeling approaches capable of reproducing realistic pedestrian behaviors across diverse environmental and social contexts. Such models enable early-stage and pre-design validation of pedestrian infrastructure, as well as systematic pedestrian flow and safety analysis.

The present research addresses key limitations in calibration accuracy, agent heterogeneity, and environmental adaptability by developing a **data-driven agent-based pedestrian simulation framework** that integrates empirical trajectory data with computational modeling. The framework combines **deep learning–assisted calibration** with **Particle Swarm Optimization (PSO)**, employing a **neural network–based surrogate model** to substantially reduce the computational cost of repeated simulations while preserving calibration fidelity. Agent behaviors, including speed modulation and directional changes, are dynamically adjusted based on local environmental conditions inferred from real-world trajectory data, enabling the capture of complex, context-dependent motion patterns.

Validation using high-resolution pedestrian trajectory datasets demonstrates the model’s ability to reproduce key emergent phenomena, including environment-aware adaptation, bottleneck formation, and collision avoidance, thereby confirming its suitability for urban design evaluation, space utilization analysis, and crowd safety management.

The contributions of this study are threefold: (i) a **local environment–conditioned calibration and validation strategy** that enhances behavioral diversity and realism; (ii) a **surrogate–**

assisted calibration pipeline that accelerates parameter estimation without sacrificing accuracy; and (iii) a **scalable and extensible simulation framework** that supports varying numbers of agents, adapts easily to diverse environment layouts, and accommodates scalable parameter configurations.

Overall, this study advances pedestrian simulation by unifying data-driven learning, environment-aware agent parameterization, and efficient calibration, providing a foundation for future extensions to group-level interactions of pedestrians in real-world environments.

Acknowledgments

First and foremost, I would like to express my heartfelt gratitude to my supervisor, **Professor Hiroshi Morita**, for his unwavering support throughout my Ph.D. studies and related research. His patience, motivation, and profound knowledge have been invaluable to me. I am deeply grateful for his wise guidance, which continually encouraged me even during the most challenging moments of my research. I could not have asked for a better advisor and mentor for my doctoral journey.

I would also like to extend my sincere appreciation to all the professors, colleagues, and friends at the Systems Engineering Laboratory, Department of Information and Physical Sciences, Graduate School of Information Science and Technology, the University of Osaka—especially Associate Professor Yutaro Yamaguchi—for their kind and constant support, helpful discussions, and warm encouragement.

My special thanks go to Professor Hideyuki Suzuki and Professor Yusuke Ogura, who kindly served as members of my thesis committee and provided valuable insights and constructive feedback that greatly improved my research.

I am also grateful to the Fellowship Program of the University of Osaka for providing financial support for my research and for enabling me to live and study in Japan during my Ph.D.

Finally, and most importantly, I would like to express my deepest appreciation to my family—my parents, my wife, and my two lovely daughters—for their unwavering love, understanding, and spiritual support throughout my Ph.D. journey and in all aspects of my life.

Contents

1	Introduction	1
1.1	Research Background	1
1.2	Purpose and Scope of the Study	2
1.3	Research Contributions	5
1.4	Thesis Organization	5
2	Literature Review	8
2.1	Foundations of Agent-Based Modeling	8
2.2	Pedestrian Dynamics Research Approaches	10
2.2.1	Empirical Approaches	11
2.2.2	Modeling Approaches	11
2.2.3	Extended Modeling Paradigms in Pedestrian Dynamics	13
2.3	Data-Driven and Hybrid Approaches in Pedestrian Simulation	14
2.4	Applications of Agent-Based Pedestrian Simulation	15
2.5	Summary and Research Motivation	17
2.5.1	Research Motivation	18
3	Methodology	19
3.1	Proposed Framework	19
3.1.1	Data Pipeline	21
3.1.2	Model Calibration	25
3.1.3	Model Execution	31
3.1.4	Quality Assessment	36

3.2	Dataset	39
3.3	Particle Swarm Optimization	42
3.3.1	Overview of PSO	42
3.3.2	Baseline Calibration using Standard PSO	44
4	Implementation and Results	46
4.1	Data and Surrogate Model Preparation	46
4.1.1	Data Collection and Preprocessing	46
4.1.2	Variable Experimental Instances	48
4.2	Experimental Setup and Environment	53
4.2.1	Computational Environment	53
4.2.2	Simulation Environment and Input Integration	54
4.3	Main Results: Model Calibration and Validation	55
4.3.1	Model Calibration Performance	56
4.3.2	Overall System Validation Results	60
4.4	Systematic Sensitivity and Experimental Design	67
4.4.1	Results and Discussion	69
4.4.2	Trade-Off Analysis: Runtime vs Quality	71
5	Discussion	73
5.1	Relating PSO-Tuned Parameters to Agent-Based Model Behavior	73
5.2	Extending the Framework Toward Single and Group Pedestrian Modeling	75
5.2.1	Preliminary Observations and Analytical Potential	76
6	Conclusion and Future Work	80
6.1	Achievements	80
6.2	Future Potential Extensions	81
	Bibliography	84

List of Figures

3.1	Overview of the proposed framework	19
3.2	Feature extraction based on local environment conditions.	24
3.3	Overview of the ATC environment layout, showing spatial boundaries, pillars, and exits.	41
4.1	Green circle markers (O) indicate valid trajectory points used in the experiment, while gray cross markers (X) represent out-of-bound points excluded.	47
4.2	Predicted vs. actual velocity.	53
4.3	All agents are initialized around the own starting point.	54
4.4	Agents are moved through the environment at step 75	55
4.5	Calibration runtime comparison.	56
4.6	Calibration accuracy vs. number of agents , evaluated using DTW-based fitness.	58
4.7	Calibration accuracy (fitness function reduction) vs. PSO iteration for 100 agents.	59
4.8	Visualization of a simulated agent’s trajectory, orientation, and interaction dy- namics.	60
4.9	Validation accuracy for different metrics and agent counts	61
4.10	Error reduction and distribution for 3 similarity metrics (DTW, Fréchet, and EDR) with heterogeneous tuning using 500 agents and 500 samples.	62
4.11	Scatter plot comparing real and simulated speeds for 200 agents.	64
4.12	Velocity-based spatial utilization heatmaps of real pedestrian trajectories at two spatial resolutions (100 cm and 50 cm grids).	65
4.13	Absolute difference heatmap of mean pedestrian velocities between real and simulated data (50 cm grid).	67

4.14	Pareto front of calibration runs.	71
5.1	Sample flock classification results of pedestrian trajectories.	76
5.2	Interaction Dashboard (Number of data points = 60).	78

List of Tables

2.1	Comparison of classical pedestrian dynamics models, associated studies, and key characteristics.	12
3.1	Feature groups with agent-specific data and local conditions.	24
3.2	Summary of dataset statistics and environmental information	40
3.3	Comparison of dataset statistics between Wednesday and Sunday recordings. . .	41
4.1	Summary of trajectory record statistics for selected agent subsets.	47
4.2	Illustrative examples of experimental instance structures.	49
4.3	General set of input features and output label used in the surrogate model. . . .	51
4.4	Training and validation RMSE reduction for different numbers of agents. Early stopping indicates the best-performing epoch.	52
4.5	Comparison of calibration accuracy between standard PSO and deep learning–assisted PSO across different agent counts.	57
4.6	Detailed sensitivity analysis summary (100-agent configuration).	70
1	Summary statistics of daily ATC pedestrian trajectory data. Each file corresponds to one day of observation (Part 1, 43 files).	90
2	Summary statistics of daily ATC pedestrian trajectory data. Each file corresponds to one day of observation (Part 2, 49 files, continued).	91
3	Detailed list of sensitivity analysis experiments (Part 1: Instances 1–48).	93
4	Detailed list of sensitivity analysis experiments (Part 2: Instances 49–57)	94

Chapter 1

Introduction

1.1 Research Background

The rapid pace of urbanization means that cities are becoming increasingly dense and complex. According to the World Bank [1], nearly 70% of the global population is expected to live in urban areas by 2050, up from approximately 56% in 2020 by UN Trade and Development (UNCTAD) [2]. This unprecedented demographic shift places enormous pressure on urban infrastructure, particularly pedestrian facilities such as walkways, intersections, public squares, and transit hubs. The development and maintenance of safe, efficient, and inclusive pedestrian infrastructure are therefore becoming a central priority for sustainable urban growth. In dense urban centers, pedestrian flow can reach up to 4–5 persons per square meter before movement slows significantly, highlighting the need for careful space allocation (Dirk Helbing [3]).

Even in hazardous and emergency conditions, pedestrians remain among the most vulnerable road users, accounting for nearly 21% of global traffic fatalities worldwide (World Health Organization [4]). In such contexts, pedestrian behavior and movement patterns play a decisive role in shaping urban mobility outcomes, including traffic efficiency, crowd management, and evacuation effectiveness. Thus, a deep understanding of pedestrian dynamics is crucial for designing infrastructure that not only ensures safety but also enhances the overall experience of moving through a city. Factors such as crowd density, spatial constraints, environmental context, and social interactions can all affect walking speed, trajectory, and route choice. For instance,

average walking speeds in urban environments range from 1.2 to 1.5 m/s under uncongested conditions, but can decrease by over 50% in high-density crowds (Moussaïd et al. [5]). Research on pedestrians dynamics is aimed at enabling planners, policymakers, and engineers to anticipate pedestrian behavior under varying environmental and social conditions, and to design interventions that improve flow, reduce congestion, and prevent accidents.

Beyond infrastructure planning, pedestrian dynamics research contributes to broader societal goals such as accessibility, inclusivity, and resilience. Well-designed pedestrian environments support the mobility of elderly individuals, people with disabilities, and families with young children, promoting social equity and public health (Litman Todd [6]).

By leveraging integrated digital twin frameworks and smart city sensing infrastructures, it is possible to monitor and analyze pedestrian flows in real time, predict congestion points, and optimize urban mobility. Using a modular, fully integrated model that combines multi-source data from IoT sensors, cameras, and environmental inputs, urban managers can simulate different scenarios, evaluate the impact of design interventions, and improve both safety and efficiency. Using such a model enables dynamic decision-making for crowd management, event planning, and emergency response, enhancing the resilience and adaptability of urban spaces (S. Xia et al [7]).

1.2 Purpose and Scope of the Study

The primary purpose of this study is to develop a **data-driven computational framework for agent-based pedestrian simulation** that balances theoretical rigor with practical applicability. The study focused on enhancing the realism and reliability of simulations through innovative calibration techniques, incorporation of agent heterogeneity, and validation against empirical data.

To achieve these, the thesis work was guided by the following **research questions**:

- **Research Question 1:** How can data-driven methods be effectively utilized to model, analyze, and simulate pedestrian dynamics?
- **Research Question 2:** In what ways can deep learning or machine learning techniques

assist in the parameter calibration of agent-based simulations to improve accuracy and computational efficiency?

- **Research Question 3:** How can agent heterogeneity be incorporated into the modeling process to enhance the realism of pedestrian dynamics?

Addressing these research questions involves navigating several **research challenges**:

Pedestrian movement modeling usually considers not only “where” pedestrians move, but more importantly “how” they move within their environment. A key challenge lies in identifying the relevant spots of the local environment—such as corners, walls, dense areas, landmarks, interactions with other agents, temporal dependencies, and group dynamics—in order to guide movement decisions at each moment. Capturing these situational factors is essential for reproducing realistic pedestrian trajectories.

- **Model Calibration:** Obtaining well-tuned parameters is necessary to reproduce real-world space–time patterns. However, calibration is computationally demanding and sensitive to data quality.
- **Social and Psychological Factors:** Pedestrian dynamics are influenced by cultural differences, social norms, group behavior, and individual decision-making. Capturing these factors within a simulation model introduces additional complexity.
- **Validation:** Ensuring that the model faithfully represents reality requires rigorous validation against empirical trajectory data. This is often challenging due to the inherent variability and complexity of pedestrian behavior.
- **Granularity of Modeling:** Pedestrian dynamics can be studied at multiple levels—system-wide, group-based, and individual-based. Choosing the appropriate level of granularity is non-trivial and directly affects the accuracy, complexity, and computational cost of the model.
- **Model Transparency and Interpretability:** As pedestrian models become increasingly complex, ensuring transparency in how movement decisions are generated is critical. Black-box or overly abstract models may achieve high predictive accuracy but provide limited insight into the underlying behavioral mechanisms, making it difficult to explain,

validate, or trust simulation outcomes—particularly for safety-critical and policy-driven applications.

Taken together, these challenges highlight the need for a modeling framework that balances behavioral realism, computational feasibility, and interpretability. Rather than attempting to address all aspects of pedestrian dynamics simultaneously, it is essential to define a focused scope that prioritizes key factors most relevant to understanding and reproducing observed movement behavior. By clearly delimiting the scope, the thesis aims to provide meaningful insights into pedestrian dynamics without sacrificing methodological clarity or practical applicability.

The scope of this thesis is as follows:

- The study proposes a **data-driven agent-based pedestrian simulation framework** that integrates empirical observations at both the initialization and calibration stages. This framework was designed to bridge the gap between microscopic pedestrian dynamics research and data-centric modeling approaches, enabling realistic simulation of individual movements and collective flow patterns observed in real environments.
- This thesis highlights the role of **deep learning–assisted calibration** to accelerate and enhance parameter tuning within an agent-based system. By employing neural network–based surrogate models, the framework reduces computational cost while maintaining high accuracy in reproducing complex pedestrian behaviors such as acceleration, deceleration, and directional changes under varying environmental constraints.
- This work incorporates **agent heterogeneity** to capture variability in pedestrian characteristics and decision-making. Each agent is parameterized based on empirical distributions of traits such as walking speed, height, and responsiveness to surrounding agents and obstacles. This heterogeneity allows the simulation to reflect the diversity of pedestrian dynamics, from solitary walkers to group movements and crowd interactions.
- The framework is calibrated and validated using **real-world pedestrian trajectory data**, with the intention of reproducing spatial and temporal patterns of pedestrian flow. The analysis particularly addresses behavioral phenomena observed in pedestrian dynamics studies—such as lane formation, bottleneck effects, and collision avoidance—to ensure the

simulated results align with empirical observations and established theoretical findings.

1.3 Research Contributions

This thesis makes several key contributions to the field of **microscopic, individual-based agent-based pedestrian simulation**, with a particular emphasis on modeling pedestrian behavior at the level of individual agents and their local interactions within complex environments:

- **Local condition-based calibration and validation:** A novel calibration and validation methodology is proposed that selects parameters based on local environmental conditions at each step of the simulation. This enables agent-level heterogeneity by assigning individualized parameters that better reflect realistic pedestrian behavior.
- **Integration of Particle Swarm Optimization (PSO) with deep learning-assisted surrogate modeling:** A hybrid calibration approach is introduced, combining the global search capability of traditional optimization techniques such as PSO with the computational efficiency of deep learning surrogates. This integration significantly reduces runtime while maintaining calibration quality.
- **Enhanced realism through heterogeneous agents:** By incorporating agent heterogeneity into the modeling process, the framework captures the variability of pedestrian dynamics across individuals and groups, leading to more realistic and reliable simulations.
- **Practical applicability for real-world scenarios:** The proposed framework improves both the accuracy and efficiency of agent-based pedestrian simulation, making it suitable for real-world applications such as infrastructure planning, design validation, and space utilization studies.

1.4 Thesis Organization

The remainder of this thesis is organized as follows:

- **Chapter 2: Literature Review**

This chapter reviews the related research in pedestrian dynamics and modeling. It groups

the literature according to several themes, including pedestrian modeling approaches (both traditional and recent), data-driven calibration strategies, various applications of pedestrian dynamics, and agent-based modeling itself. The chapter also discusses the challenges and opportunities in the field, with a particular focus on model calibration. In addition, it examines prior studies that applied optimization techniques, such as Particle Swarm Optimization (PSO), and other traditional search algorithms to the calibration of agent-based models.

- **Chapter 3: Methodology** This chapter introduces a proposed methodology and establishes the technical foundation of the study. It begins with the dataset description and the use of Particle Swarm Optimization (PSO) as a baseline calibration method. The chapter then presents the proposed framework, composed of four core modules: **Data Pipeline**, **Model Calibration**, **Model Execution**, and **Quality Assessment**.

The **Data Pipeline** module handles the preprocessing and feature extraction of trajectory data, ensuring consistency and reliability. The **Model Calibration** module integrates PSO and deep learning–assisted surrogate modeling to estimate heterogeneous agent parameters efficiently. The **Model Execution** module describes the simulation process, where calibrated agents interact within a virtual environment under local behavioral and environmental constraints. Finally, the **Quality Assessment** module defines evaluation metrics and validation procedures to compare simulated and empirical pedestrian dynamics.

Together, these modules form an integrated framework linking empirical data, model calibration, and simulation-based validation for realistic pedestrian dynamics modeling.

- **Chapter 4: Implementation and Results**

This chapter presents the full implementation of the proposed data-driven framework and reports the key experimental findings. It begins by describing the data collection and preprocessing procedures used to construct the real-world pedestrian trajectory dataset, followed by the preparation and training of the surrogate models incorporated into the proposed framework.

It then outlines the experimental design, including the definition of multiple experimental instances and the full description of the computational and simulation environments. The main results are presented through systematic model calibration and validation against real pedestrian data, accompanied by performance evaluations across different simulation scales with varying numbers of agents.

Finally, a comprehensive sensitivity analysis is conducted to examine how different design choices and parameter variations influence model accuracy, behavioral fidelity, and computational efficiency.

- **Chapter 5: Discussion**

This chapter discusses the broader implications of the findings. It examines the impact of local condition encoding on simulation quality and analyzes the relationship between PSO-tuned parameters and emergent behaviors in an agent-based model. The discussion covers both theoretical contributions and practical insights.

Furthermore, the framework presents a flexible foundation for extending the current model to capture *Single vs. Group* pedestrian dynamics. By incorporating mechanisms that distinguish between individual and collective movement patterns, future work can explore how group-level coordination, shared goals, and intra-group spacing influence the emergence of macroscopic flow behaviors. This extension would enhance the model's applicability to real-world crowd scenarios where both single and group pedestrians coexist and interact dynamically.

- **Chapter 6: Conclusion and Future Work**

The final chapter summarizes the main contributions of the study and reflects on its limitations. It also outlines directions for future work, including opportunities to extend the framework to more complex scenarios and to integrate additional social and psychological factors into the model.

Chapter 2

Literature Review

This chapter reviews related research in pedestrian dynamics and modeling. The literature is organized according to several themes, including classical and modern pedestrian modeling approaches, agent-based modeling and its theoretical foundations, data-driven calibration strategies, and optimization techniques used for model calibration. The chapter also highlights the challenges and opportunities in the field, with a particular focus on improving the realism and reliability of simulations through the integration of empirical data and computational methods.

2.1 Foundations of Agent-Based Modeling

Agent-Based Modeling and Simulation (ABMS) provides a bottom-up framework for modeling complex systems, in which macroscopic patterns emerge from local interactions between heterogeneous, autonomous agents. ABMS emphasizes agent autonomy, local interaction rules, bounded rationality, and emergent phenomena (Macal and North [8]). It enables researchers to represent individuals as decision-making entities whose collective interactions lead to emergent system-level outcomes. The ABMS paradigm has been widely adopted in computational social science, economics, and urban studies to explore adaptive and non-linear dynamics in complex systems.

The conceptual origins of agent-based modeling (ABM) can be traced back to the 1970s and 1980s, when researchers first began exploring how simple individual rules could generate complex collective outcomes. One of the earliest examples is Thomas Schelling's segregation

model [9], introduced in his seminal 1971 work, *Dynamic Models of Segregation*. Although initially implemented with coins and graph paper rather than computers, Schelling’s model embodied the core principles of ABM: autonomous agents interacting locally within a shared environment to produce emergent, system-level patterns. In the following decades, the conceptual and computational foundations of ABM were further formalized. Holland and Miller [10] advanced the field by introducing the notion of artificial adaptive agents, describing how boundedly rational agents could adapt through learning and evolution in economic systems. Their work helped define the modern understanding of “agents” as adaptive, interactive entities, laying the groundwork for the computational agent-based frameworks that emerged in the 1990s and beyond.

The formalization of agent-based modeling has seen significant advancement through the development of standardized modeling protocols such as the ODD (Overview, Design concepts, and Details) and ODD+2D frameworks, which promote transparency, reproducibility, and systematic mapping between empirical data and model structure (Grimm et al. [11]). These protocols provide a clear structure for describing agent behaviors, interactions, and environmental settings, enabling consistent interpretation and replication of complex models across studies. They are particularly valuable when integrating real-world pedestrian trajectory data, as they ensure that model assumptions, behavioral rules, and parameters are explicitly documented, which is important for comparability and validation of simulation outcomes.

Despite its versatility, ABMS tives rise to several inherent challenges. For example, the high dimensionality of agent interactions can generate complex emergent behaviors that complicate calibration and validation against empirical data (Li An et al. [12]).

Conte and Paolucci [13] provided a critical survey of the field of agent-based modeling (ABM) and its role within the broader domain of computational social science (CSS). They argued that although ABM has demonstrated strong capabilities for modeling heterogeneous interacting agents and reproducing emergent phenomena, its full generative potential remains underutilized because it is often constrained by minimalistic or ad hoc rule-based formulations. The authors identified key challenges in the field that included achieving scalability, calibrating and validating models against empirical data, and realizing consistency of micro-to-macro link-

ages in the simulation design. Moreover, they distinguished three complementary variants of CSS—deductive, generative, and complex—and advocated for an interdisciplinary, multilevel modeling approach that integrates data-driven, theory-driven, and simulation-driven perspectives. Ultimately, they argued that advancing ABM within the CSS domain would require richer agent representations, stronger theoretical grounding, and tighter integration with empirical and quantitative methodologies.

There have been a number of important early and formative achievements in the field of agent-style pedestrian modeling. Craig Reynolds’ Boids [14], which inspired later agent-based approaches, uses distributed agent rules for collective motion. Dirk Helbing and Péter Molnár’s Social Force Model [15], which became a foundational individual-level model for pedestrian dynamics, drives pedestrian motion by “social force.” Several influential cellular automaton / agent microsimulation papers appeared in 2000 and 2001, including especially those of Blue and Adler [16] and Burstedde et al. [17]; these papers introduced floor-field and CA-based microsimulations, which are now widely used for pedestrian flow and evacuation studies. The above work collectively shaped the move from conceptual agent rules toward computational ABMS and microsimulation methods for pedestrian flow.

2.2 Pedestrian Dynamics Research Approaches

Traditional research approaches to pedestrian dynamics generally fall into two broad categories: (1) empirical studies, which rely on direct observation and controlled experiments to document real-world behavior, and (2) modeling approaches, which use mathematical and computational tools to represent, analyze, and simulate pedestrian movement.

Empirical insights form the behavioral foundation of pedestrian dynamics, and then modeling approaches translate these insights into formal structures that can be simulated, tested, and applied to real-world design problems. The following subsections summarize these two streams of research.

2.2.1 Empirical Approaches

Early and foundational work in pedestrian dynamics was rooted in direct empirical investigation, where researchers observed and quantified human movement to identify behavioral patterns and establish fundamental flow relationships.

- **Direct Observation and Field Studies:** Researchers conducted on-site observations of movement on sidewalks, at intersections, around stations, and in other urban spaces to gather data on speed, density, flow, and avoidance behavior. Manual counting, sketches, and rudimentary measurement tools were used before automated sensors became available.
- **Photographs, Video Recording, and Time-Lapse Studies:** Adoption of photographic and time-lapse recording enabled detailed post-analysis of crowd behavior. Sequential images reveal spatial patterns such as density fluctuations, segregation of directional flows, and localized congestion. These techniques supported the early empirical laws, including speed–density and flow–density curves.
- **Controlled Experiments:** Laboratory-style experiments provide controlled conditions for examining specific phenomena, such as bottleneck flow, merging streams, evacuation dynamics, and collective behaviors under varying densities. Such experiments became the standard benchmarks for validation when the subsequent simulation models were developed.
- **Surveys and Interviews:** Qualitative methods capture the perceptual and cognitive aspects of pedestrian behavior. Questionnaires and interviews informed the early understanding of route choice, comfort, perceived safety, crowding tolerance, and environmental preferences, complementing quantitative field data.

2.2.2 Modeling Approaches

Building on empirical evidence, a wide spectrum of modeling paradigms for representing pedestrian behavior have been developed at varying levels of abstraction. Classical approaches include

force-based models, grid-based cellular automata, velocity-driven formulations, and rule-based behavioral systems, with each offering different trade-offs between realism, interpretability, and computational efficiency.

Table 2.1 summarizes representative studies and highlights the key characteristics of these foundational modeling approaches, which have shaped modern pedestrian simulation and informed the development of hybrid and data-driven models.

Table 2.1: Comparison of classical pedestrian dynamics models, associated studies, and key characteristics.

Model	Study	Description
Social Force Model (SFM)	Helbing and Molnár [15]; Helbing et al. [18]; Moussaïd et al. [19]	Pedestrians are modeled as self-driven particles with attractive and repulsive forces. Captures continuous motion and emergent patterns such as lane formation. Requires careful calibration and may yield unrealistic accelerations in dense crowds.
Cellular Automata (CA)	Zhang et al. [20]; Li et al. [21]; Weng et al. [22]	Discrete grid-based movement governed by local transition rules. Highly efficient and suited to high-density flows. Enhancements such as potential fields and event-driven updates improve realism. Movement discretization can reduce trajectory smoothness.
Velocity-Based Models	Van den Berg et al. [23]; Xu et al. [24]; Hu et al. [25]; Zhang et al. [26]	Computes collision-avoiding velocities directly, producing smooth trajectories. Effective for robotics, computer graphics, and pedestrian simulation. Extensions handle bottleneck flows and direction strategies, but social/group behaviors are often simplified.
Rule-Driven Approaches	Bonabeau [27]; Al-Kodmany [28]	Models pedestrian actions via explicit rules or heuristics (e.g., following, lane formation, priority decisions). Flexible for representing cognitive or social behaviors, though less suited to dense, complex interactions.

These traditional approaches form the foundation of pedestrian modeling, being tools for both microscopic and macroscopic simulation. In recent years, advances in data collection and computational methods have led to the development of data-driven and network-based models, which we discuss in the next section.

2.2.3 Extended Modeling Paradigms in Pedestrian Dynamics

- **Agent-Based Models (ABMs):** General framework in which pedestrians are modeled as autonomous agents with individual characteristics, decision-making, and local interactions. SFM and CA can be seen as specific ABMs, but more general ABMs allow heterogeneous behaviors.
- **Continuum or Fluid-Dynamic Models:** Treat crowds as continuous media, similar to fluids. Useful for simulating large-scale, high-density flows efficiently.
- **Network or Graph-Based Models:** Pedestrian movements are represented as flows on networks comprising nodes correspond to locations and edges corresponding to paths. Commonly used for evacuation planning or large-facility navigation.

Modern pedestrian simulations increasingly integrate elements from multiple paradigms to balance realism, scalability, and computational efficiency. Recent studies have emphasized agent heterogeneity, adaptive decision-making, and group-level dynamics (e.g., Saeed et al. [29]), incorporating real trajectory data to allow agents to respond to local density, environmental constraints, and social interactions in a more nuanced manner.

Several comprehensive reviews have synthesized progress across microscopic, macroscopic, and hybrid modeling approaches. Shanwen et al. [30] highlighted the evolution of crowd simulation techniques and the ongoing challenges of achieving behavioral realism while maintaining computational tractability. Papadimitriou et al. [31] underscored the importance of human factors, such as perception, attention, and decision-making, in shaping pedestrian motion, emphasizing aspects that purely physical or rule-based models often overlook.

The above reviews illustrated the breadth and versatility of pedestrian modeling frameworks, from behavioral analysis to intelligent transportation systems. These reviews also laid the groundwork for the emergence of data-driven and hybrid modeling paradigms, which aim to bridge empirical trajectory data with simulation models to achieve greater realism and adaptability.

2.3 Data-Driven and Hybrid Approaches in Pedestrian Simulation

Recent advances in sensing technologies, large-scale trajectory datasets, and machine learning methods have enabled the emergence of data-driven and hybrid pedestrian simulation approaches. Unlike classical models, which rely mainly on predefined behavioral rules or physical analogies, data-driven models learn motion patterns directly from empirical trajectories. These approaches can capture subtle behavioral variations, heterogeneous responses to local conditions, and complex interactions that are difficult to encode manually. Hybrid methods combine these data-driven components with traditional mechanisms, allowing models that leverage both physical interpretability and data adaptivity. Together, these techniques represented a significant evolution in pedestrian modeling and opened new possibilities for realism and predictive accuracy.

Machine learning techniques have been applied to infer pedestrian behavior patterns from trajectory datasets (George et al. [32]; Weili Wang [33]).

Recent advances have incorporated data-driven and neural network approaches to enhance behavioral realism. For example, Xiao et al. [34] proposed a data-driven neural network model for simulating pedestrian movement that captures nonlinear behavioral responses more effectively than traditional rule-based models.

Additionally, Butts et al. [35] introduced a data-driven agent-based modeling framework for animal movement, emphasizing exploratory data analysis as a foundation for agent rule design. Jinghui et al. [36] conducted a comprehensive survey of data-driven crowd modeling techniques, discussing their implications for ABMS. Jiang and Jia [37] demonstrated how human movement patterns can emerge from underlying street structures through an agent-based approach, establishing a link between spatial configuration and emergent collective behavior.

Hybrid approaches combine ABMS with optimization and surrogate modeling. Particle Swarm Optimization (PSO) has been widely applied to calibrate agent parameters (e.g., Michels et al. [38]; Du et al. [39]). Surrogate-assisted optimization reduces computational costs by approximating expensive simulation runs (Jin et al. [40]). Reinforcement learning-

based calibration approaches allow adaptive real-time tuning of model parameters, improving simulation fidelity in dynamic environments (Yang et al. [41]). These methods address major ABMS challenges, including stochastic variability, calibration complexity, and computational scalability.

Several recent studies demonstrated frameworks that integrate agent-based modeling with data-driven and optimization techniques. Jamali and Lazarova-Molnar [42] proposed a framework for integrating data-driven methods into agent-based modeling. Their approach highlights how real-world data can automate model construction to support continuous updates. Their prior work also demonstrated how data-driven techniques can capture empirical patterns for model development.

Zhang et al. [43] applied a data-driven ABMS framework in which individual decision rules are learned using machine learning and embedded into a multi-agent system. Their model was validated using holdout adoption sequences and successfully forecasted both individual and aggregate rooftop solar adoption in San Diego County. Compared to a traditionally calibrated ABM, the data-driven model, estimated via maximum likelihood, achieved substantially better predictive performance. They also used the model to explore improved incentive structures, showing that simple heuristic searches can yield more effective policies and that providing free systems to low-income households is particularly impactful.

2.4 Applications of Agent-Based Pedestrian Simulation

Agent-based modeling has been widely applied to analyzing and simulating pedestrian behavior across diverse real-world settings, from architectural design evaluation to behavioral prediction and safety management.

For urban and architectural research, Liang Chen [44] provided a comprehensive overview of ABMS applications appearing in spatial and design-oriented studies. In that overview, the value of ABMS for examining emergent movement patterns and assessing the performance of built environments was emphasized. Crooks et al. [45] discussed future directions for geographical agent-based models, emphasizing challenges in integrating spatial data infrastructures, behav-

ioral realism, and multi-scale urban representation. David et al. [46] further extended these ideas to architectural contexts, reviewing ABMS applications in building design, evacuation planning, and occupant behavior analysis, with particular attention to simulation–BIM integration. Kostas [47] proposed an agent-based framework for modeling public space use, and demonstrated how spatial configuration, temporal rhythms, and user heterogeneity jointly shape emergent occupancy and movement patterns.

At the micro level, Jiang [37] showed how underlying street structures influence pedestrian route choices and collective flow patterns, linking individual decisions to spatial network characteristics.

Voloshin et al. [48] proposed an optimization-based calibration framework for micro-level pedestrian agent simulations in public spaces. Their approach automatically estimates behavioral parameters by minimizing the discrepancy between simulated and observed pedestrian trajectories, which improves model realism without manual tuning. Their study demonstrated that optimization-driven calibration can significantly enhance predictive accuracy and reliability in pedestrian ABMS.

Papadimitriou et al. [31] examined perceptual and decision-making processes underlying walking and crossing behavior from a human-factors perspective, offered insights that inform more behaviorally grounded agent rules.

Among applied domains, Jahedul et al. [49] developed an agent-based simulation framework for airport environments that incorporates social distancing strategies. Their work demonstrated how ABMS can support operational safety and facility management under dynamic public health constraints. Manon et al. [50] explored pedestrian–vehicle interactions in shared spaces through an agent-based model for predicting pedestrian trajectories near autonomous vehicles. This example illustrates the growing relevance of ABMS in emerging mobility systems. Shanwen et al. [30] reviewed crowd simulation methodologies more broadly and emphasized how ABMS bridges individual-level behavioral representations with large-scale crowd dynamics.

The above demonstrates the versatility of ABMS across urban planning, architectural design, transportation hubs, and safety analysis, as well as illustrating the ongoing convergence between spatial analytics, behavioral modeling, and data-driven simulation. We can conclude from these

a need for modeling frameworks that can incorporate empirical trajectory data, capture local interactions, and adapt to heterogeneous pedestrian behaviors.

2.5 Summary and Research Motivation

This chapter establishes ABMS as one of the fundamental conceptual foundations for pedestrian dynamics, reviews both classical modeling paradigms and modern data-driven extensions, and highlights the growing importance of empirically grounded simulation.

Despite significant progress in pedestrian dynamics research, several key gaps remain:

- **Limited calibration accuracy against real-world data:** Many models rely on heuristic or scenario-specific parameter tuning, resulting in weak alignment with empirical pedestrian trajectories.
- **Simplified or missing group interaction modeling:** Group-level behaviors and social interactions are often oversimplified or neglected, limiting the realism of simulated pedestrian dynamics.
- **Insufficient representation of agent heterogeneity:** Individual differences in speed, reaction, and movement strategies are frequently ignored or modeled using uniform parameters.
- **High computational cost in large-scale simulations:** Detailed behavioral models often incur prohibitive computational costs when applied to dense or large-scale environments.
- **Limited linkage between microscopic behavior and macroscopic space–time patterns:** While many models focus either on individual-level interactions or aggregate flow characteristics, fewer studies systematically examine how microscopic modeling and calibration choices influence emergent macroscopic phenomena such as congestion formation, space utilization, and temporal flow patterns.
- **Validation methods not aligned with ground truth:** Many validation approaches rely on qualitative assessment or aggregate statistics, rather than direct comparison with high-resolution trajectory data.

2.5.1 Research Motivation

Recent data-driven and hybrid modeling approaches have demonstrated promising pathways for addressing these challenges by integrating empirical observations, machine learning techniques, and optimization-based calibration. In particular, recent advancements have emphasized **individual-based calibration**, where agent behaviors are directly informed by empirical trajectory data, enabling data-fed agents to adapt their movement strategies according to local environmental and interaction conditions. Such approaches support condition-aware parameterization, richer behavioral realism, and improved predictive performance across diverse spatial and temporal contexts.

However, despite these advances, balancing behavioral fidelity with computational efficiency—especially in large-scale, high-density, or scenario-based simulations—remains an open research problem. Moreover, many existing methods lack systematic validation frameworks that explicitly connect agent-level calibration to emergent macroscopic space–time patterns.

Building upon these insights, the motivation for this research is to develop a **data-driven, individual-based calibration and validation framework** that bridges empirical pedestrian trajectory data and agent-based simulation. The framework proposed in Chapter 3 leverages optimization, surrogate modeling, and context-dependent parameterization to explicitly address the identified research gaps. By incorporating empirically grounded agent heterogeneity and scalable calibration strategies, the proposed approach aims to enhance both microscopic behavioral fidelity and macroscopic pattern realism, while maintaining computational tractability. This enables practical applications in urban design validation, safety assessment, and space utilization analysis.

Chapter 3

Methodology

3.1 Proposed Framework

The proposed framework integrates a deep learning surrogate model with the Particle Swarm Optimization (PSO) algorithm within an Agent-Based Modeling and Simulation (ABMS). The framework begins with trajectory data, where each record contains a unique pedestrian identifier and corresponding spatiotemporal information, including timestamps, positions, velocities and other states. This hybrid integration effectively addresses the computational challenges of parameter calibration while enabling efficient and realistic simulations of pedestrian dynamics in complex indoor environments.

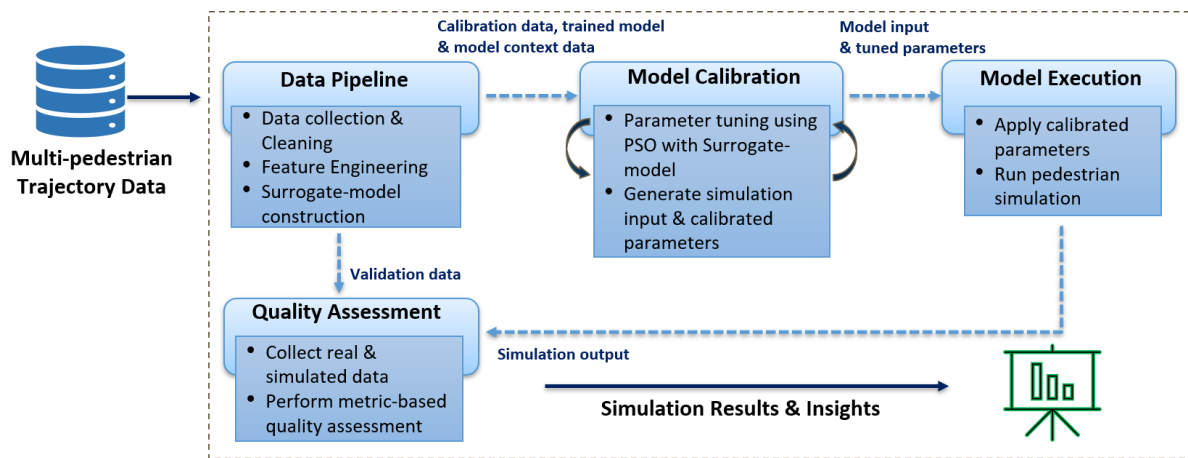


Figure 3.1: Overview of the proposed framework

As depicted in Figure 3.1, the framework consist of four main modules: (1) Data Pipeline,

(2) Model Calibration, (3) Model Execution, and (4) Quality Assessment. Each module is designed for a specific task, yet they operate in an integrated workflow to ensure accurate calibration and efficient simulation.

The overall design of the framework follows the principle of **Local Conditions + Current References + Tunable Parameters**, which serves as the foundation for agent-specific calibration.

The key novelty of this research lies in introducing tunable parameters that are conditioned on local environmental states at the individual agent level. This design is motivated by the observation that pedestrians traversing the same local environment—such as corners, narrow passages, or interaction-prone regions—tend to exhibit consistent response patterns when exposed to similar contextual and interaction conditions.

Leveraging this behavioral regularity, the proposed framework identifies and optimizes parameter sets that are specific not only to each agent, but also to the local conditions encountered along their trajectories. This enables the model to associate recurring environmental contexts with stable behavioral responses, allowing pedestrian behavior to be modeled as a function of both agent identity and situational context.

As a result, pedestrian behavior—conditioned on environmental context and neighboring interactions—can be modeled, optimized, and continuously updated within a unified framework, achieving higher behavioral fidelity while maintaining interpretability and computational efficiency.

Local conditions represent the agent’s immediate environmental context, capturing how nearby spatial and social factors influence movement decisions. In real-world pedestrian behavior, individuals constantly adapt their velocity and direction based on the surrounding environment, such as approaching walls or pillars, entering or leaving crowded areas, or interacting with nearby pedestrians. To emulate this adaptability, the model encodes such spatial interactions into *local condition features*. These features are derived from geometric proximity and dynamic states, such as the agent’s distance to obstacles, body height, relative motion direction, and surrounding agent density. Each condition is converted into a numerical or categorical *relative code*, which serves as an abstract representation of the current situation as perceived

by the agent. By learning from these encoded local conditions, the deep learning model can infer how pedestrians adjust their movement under various spatial constraints, enabling more context-aware and realistic behavioral prediction compared to models relying solely on global or averaged features.

3.1.1 Data Pipeline

This module processes real-world pedestrian trajectory data and prepares all required datasets for downstream modules, including calibration, validation, and surrogate model construction. The pipeline consists of the following stages:

- **Data Collection & Cleaning:** Raw trajectory data are first collected from the real-world sensing system. To ensure reliable inputs, the cleaning process removes inconsistencies, measurement noise, and outliers. Specifically, the environment boundaries are reconstructed using a set of line segments that form a closed polygonal region. Any trajectory points lying outside this polygon are removed to maintain strict spatial consistency. After spatial filtering, the dataset is further reduced to include only trajectories belonging to a predefined number of agents; this dataset corresponds to that of the experimental setup used to demonstrate the scalability and feasibility of the proposed framework.
- **Dataset Splitting:** The cleaned dataset is divided into two primary subsets: a calibration dataset and a validation dataset. The calibration dataset is then partitioned into three components: (i) a training set for surrogate model learning, (ii) a testing set for performance monitoring, and (iii) a PSO validation set, which is used during parameter optimization to evaluate candidate agent-parameter configurations.
- **Feature Engineering:** From the cleaned and structured trajectories, relevant features are extracted. These include agent kinematic variables (positions, velocities, and motion angles), interaction information (surrounding agents, distances, and local density), and environment-dependent factors (e.g., nearest wall distance, nearest corner distance, spot annotations). These features serve as the basis for both surrogate modeling and agent-level behavioral calibration.

- **Model Training and Testing:**

A surrogate regression model is trained using the engineered features to approximate the underlying pedestrian movement behavior. The trained model is then evaluated on the held-out testing data to ensure generalization before being integrated into the calibration and simulation modules.

Dataset Splitting

After cleaning and preparing the raw pedestrian trajectory data, the next step is constructing the datasets required for surrogate model training, calibration, and independent validation. To ensure that all environmental conditions and agent behaviors are well represented across splits, we implement a stratified per-agent data splitting procedure, which operates as follows:

- **Ensuring Complete Condition Coverage:**

Pedestrian motion depends on two categorical environmental factors: (i) the obstacle (wall and pillar) condition code, and (ii) the crowd density condition code. For each agent, all combinations of these condition codes are enumerated. If any combination is missing in the raw data, the pipeline automatically synthesizes a representative row using the following: the agent’s own mean values for available conditions, and global mean values computed across all agents for missing behavioral parameters. This guarantees that **every agent** contains a complete set of condition states prior to splitting.

- **Condition-Aware Feature Aggregation:**

For each (AgentID, Local_Environment_Codes) group, statistical summaries (e.g., mean velocity, turning behavior, acceleration percentages, relative wall/pillar angles) are computed and appended as features. These aggregated values support stratification by ensuring that splits preserve the behavioral diversity associated with different conditions.

- **Uniform Record Selection:**

To maintain consistency across agents and conditions, a fixed number of samples per condition are selected (default: `MAX_RECORDS`). This prevents agents with unusually long trajectories from dominating the training process and thereby ensures balanced

representation.

- **Hierarchical Stratified Splitting:**

The filtered dataset for each agent is split into four subsets using the following distribution:

Global validation=0.2, Model training=0.5, Testing=0.2, PSO Validation=0.1.

The splitting process is hierarchical, as follows:

1. First, a **global validation** set is separated.
2. The remaining calibration data are then split into:
 - a **training set** for surrogate modeling,
 - a **testing set** for monitoring generalization, and
 - a **PSO validation set** used during parameter optimization.

All splits are **agent-wise**, meaning each agent’s data are divided independently, ensuring:

- no cross-agent contamination,
- consistent condition coverage across all subsets, and
- fair comparison during calibration.

- **Post-Split Consistency Verification:**

After splitting, the pipeline performs condition-coverage checks. For each dataset (Train, Validation, Test, and PSO Validation), the pipeline verifies whether every agent retains all expected combinations of obstacle and crowd-density codes. Any missing patterns are reported for debugging.

The above stratified and condition-complete dataset construction ensures that all downstream processes (surrogate model training, PSO-based calibration, and final performance validation) are based on balanced, representative, and unbiased samples of pedestrian behavior.

Surrogate Model Design and Feature Extraction

To construct the surrogate model, we engineered features capturing local environmental conditions at each timestep. These include spatial proximity to walls, pillars, and surrounding agents,

which are critical inputs for predicting pedestrian velocities. Figure 3.2 illustrates the overall feature extraction process.

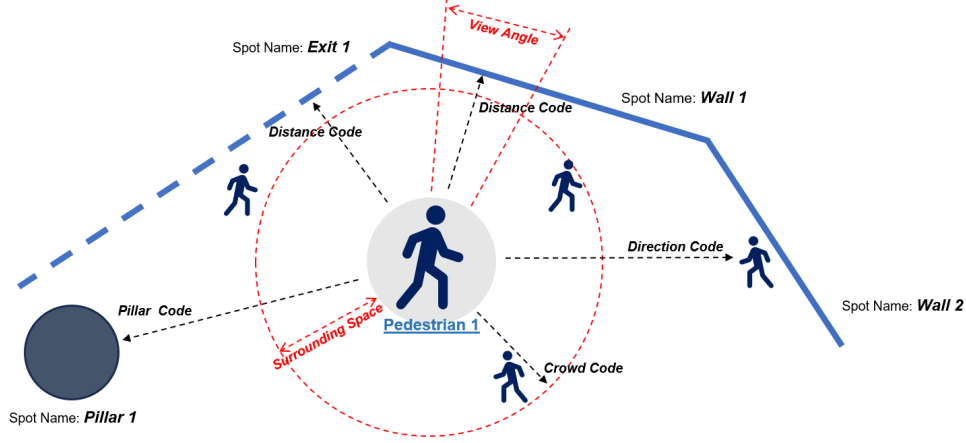


Figure 3.2: Feature extraction based on local environment conditions.

During feature engineering, several parameters were defined:

- *View Angle*: Range between -90° and 90° .
- *Surrounding Space (crowd radius)*: Relative to the agent's body size, typically proportional to height.
- *Distance-to-Wall Threshold*: Dependent on the agent's movement rate.
- *Distance-to-Pillar Threshold*: Based on vertical profile, approximately proportional to height.

Table 3.1: Feature groups with agent-specific data and local conditions.

Input (Features)	Feature Group	Description
agentID	Local Conditions	Unique identifier for each agent.
Relative Codes	Local Conditions	Encodes current environmental spots (walls, pillars, surrounding agents).
Reference Inputs	References	Step-based reference values guiding behavior.
Tunable Features	Tunable	Optimization variables adjusted by PSO.
Output (Label)	Velocity	Predicted velocity of the agent.

As described in Table 3.1, the surrogate model, trained on these features, accelerates evaluation during calibration by replacing repeated simulation calls with fast predictions. By operating

directly on locally conditioned, agent-specific tunable parameters, the surrogate preserves behavioral fidelity while significantly reducing computational cost, enabling scalable and efficient calibration in large-scale or data-intensive scenarios.

Architectural Choice and Training Considerations

For the surrogate model, we selected a deep learning architecture suitable for tabular data. This choice was motivated by several factors:

- **Tabular Nature of Data:** Each timestep is represented by a fixed set of numerical and categorical features, which aligns well with architectures like TabNet or fully connected networks optimized for tabular inputs.
- **Efficient Learning of Local Dependencies:** The model can learn how local environmental conditions interact with agent-specific attributes to influence movement decisions.
- **Scalability:** Tabular architectures enable fast training and inference across thousands of agents and timesteps, making it practical for iterative calibration using PSO.

The training procedure involves standardization of numerical features, one-hot encoding of categorical variables, and optimization using mean squared error (MSE) loss for velocity prediction. Early stopping and validation on held-out data ensure that the model generalizes well without overfitting. By using a deep learning surrogate model tailored to tabular features, we can dramatically reduce the computational cost of calibration while maintaining predictive accuracy and context-aware behavior modeling.

3.1.2 Model Calibration

In this subsection, we present our innovative deep learning-assisted model calibration method. This module performs two primary tasks:

1. **Calibration and transformation of agent-specific parameters** based on local conditions. These include:
 - Minimum and maximum walking speeds

- Acceleration and deceleration percentage changes
- Turn angle adjustment percentage
- Sensitivity to surrounding crowd dynamics

2. **Generation of simulation inputs**, which include:

- Number of agents
- Environment size (minimum and maximum coordinates)
- Environment boundary line segments with names and coordinates
- Pillars with center coordinates and radii
- Agent static properties: ID, local and global reference parameters, body diameter, initial position
- Indicated routes

Indicated route generation. To achieve realistic pedestrian movement, we extract an *indicated route* for each pedestrian from real trajectory data. An indicated route is a simplified sequence of key points that captures the pedestrian's essential movement pattern without relying on the full raw trajectory.

We employ a trajectory simplification procedure consisting of three steps:

1. Extract key points that capture major direction changes or stops.
2. Detect turning points that serve as decision locations.
3. Connect the key points to form simplified, passable routes.

Benefit: This approach preserves essential movement patterns while reducing trajectory complexity.

The deep-learning-assisted PSO algorithm is as follows.

Algorithm 1: Deep Learning–Assisted PSO for Agent Parameter Optimization

Input: Agent data $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$, Environmental conditions $\mathcal{E}$, PSO parameters (c_1, c_2, w) , Max iterations I_{max} , Number of particles P

Output: Optimized tuning parameters Θ^* for each agent

Initialize:

- Randomly initialize P particles with tuning parameters $\Theta = \{\theta_1, \theta_2, \dots, \theta_P\}$ for each agent $a \in \mathcal{A}$
- Set velocities $v_p \leftarrow 0$ for all particles
- Load surrogate model $\mathcal{M}$ trained on agent-specific data
- Set global best position $g_{best} \leftarrow \infty$
- Set individual best positions $p_{best} \leftarrow \infty$ for all particles

foreach agent $a \in \mathcal{A}$ **do**

Load agent-specific environment $\mathcal{E}_a$

for $t = 1$ **to** I_{max} **do**

foreach particle $p = 1 \dots P$ **do**

Evaluate Objective Function:

Predict velocity: $\hat{v}_p \leftarrow \mathcal{M}(a, \mathcal{E}_a, \theta_p)$

Compute fitness: $f_p \leftarrow \text{Obj}(\hat{v}_p, v_{\text{real}, a})$

Update Personal and Global Best:

if $f_p < p_{best}[p]$ **then**

$p_{best}[p] \leftarrow f_p, p_{best_pos}[p] \leftarrow \theta_p$

end

if $f_p < g_{best}$ **then**

$g_{best} \leftarrow f_p, g_{best_pos} \leftarrow \theta_p$

end

end

Update Particles:

foreach particle p **do**

$v_p \leftarrow wv_p + c_1r_1(p_{best_pos}[p] - \theta_p) + c_2r_2(g_{best_pos} - \theta_p)$

$\theta_p \leftarrow \theta_p + v_p$

end

end

end

Return: optimized parameter exported by local environmental conditions using Θ^* for each agent.

Objective Function.

The calibration problem is formulated as

$$\min_{\theta} J(\theta) \quad \text{s.t.} \quad \mathbf{g}(\theta) \leq 0, \quad (3.1)$$

where θ is the parameter set and $\mathbf{g}(\theta)$ are feasibility constraints. The objective function compares predicted and observed trajectories using similarity measures such as DTW, Fréchet

distance, or EDR (Tao et al. [51] compared five of the most important and commonly used similarity measures of trajectories):

$$J(\theta) = \frac{1}{NT} \sum_{i=1}^N \sum_{t=1}^T E(\hat{y}_i(t, \theta), y_i(t)), \quad (3.2)$$

where

- N is the number of agents,
- T is the number of time steps,
- $E(\cdot)$ is a general error function quantifying the discrepancy between the predicted and real states at each time step.
- $y_i(t)$ denotes the observed or ground-truth state obtained from empirical trajectory data, and
- $\hat{y}_i(t, \theta)$ indicates the corresponding model-predicted state generated under a parameter set θ .

Each state $y_i(t)$ and $\hat{y}_i(t, \theta)$ represent the real and predicted trajectories, respectively, of agent i at time t .

Both states can encompass multiple aspects of agent behavior, including local environmental conditions, instantaneous movement characteristics (e.g., velocity, acceleration, direction), and interactions with neighboring agents. These elements align with the structured representation adopted in this framework, comprising `local condition`, `current reference`, and `tunable parameter` components, allowing the model to capture the context-dependent dynamics of pedestrian motion across varying spatial and temporal settings.

Explanation of Algorithm 1

Algorithm Inputs The Algorithm 1 uses the following inputs:

- **Agent Data** ($\mathcal{A}$): A list of agents, each identified by a unique ID (e.g., $a_1, a_2, \dots$).
- **Environment Conditions** ($\mathcal{E}$): Conditions specific to the agent's surroundings, such as wall distances, nearby agents, or environmental features.
- **PSO Parameters:**

- c_1 and c_2 : Acceleration coefficients that control the influence of personal and global best positions on velocity updates.
 - w : inertia weight, controlling the influence of the previous velocity.
 - P : Number of particles (candidate solutions) in the PSO.
 - $I_{\max}$: Maximum number of PSO iterations.
- **Surrogate Model ($\mathcal{M}$):** A pre-trained deep learning model that predicts agent-specific behaviors based on the following:
 1. *Agent ID+Current Environment Condition*
 2. *Current References*
 3. *Tuning Parameters*

Algorithm Steps The deep-learning-assisted PSO framework involves the following steps:

1. **Initialization:** Each agent is initialized using a unique set of P particles. Each particle represents a potential solution (tuning parameters) and is assigned as follows:
 - Random initial positions within the defined parameter space.
 - Zero velocity.
2. **Fitness Evaluation:** For each particle, the fitness is computed as follows:
 - (a) Surrogate model $\mathcal{M}$ is used to predict outcomes for the particle.
 - (b) The error using the objective function defined in 3.2 is calculated.
3. **Update Personal and Global Bests:** Each particle updates its personal best position if its current fitness is better than its previous best. The global best is updated if the fitness of any particle is better than the current global best value.
4. **Velocity and Position Updates:** Particle velocities and positions are updated using the PSO equations:

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (p_{i,\text{best}} - x_i(t)) + c_2 \cdot r_2 \cdot (g_{\text{best}} - x_i(t)), \quad (3.3)$$

$$x_i(t + 1) = x_i(t) + v_i(t + 1), \quad (3.4)$$

where

- $v_i(t)$ is the velocity of particle i at iteration t ,
- $x_i(t)$ is the position of particle i at iteration t ,
- $p_{i,\text{best}}$ is the personal best position of particle i ,
- g_{best} is the global best position among all the particles, and
- r_1, r_2 are random numbers in $[0, 1]$.

5. **Stopping Criteria:** The Algorithm 1 iterates until the maximum number of iterations (I_{max}) is reached. The best parameters for each agent are selected based on the global best positions.

Agent Heterogeneity in a model calibration

We introduced agent heterogeneity by allowing each agent to have individualized parameters. Data from real-world trajectories (comprising the timestamp, agent ID, position, velocity, height, and motion angle) were used to infer distinct characteristics for each agent. Therefore, the calibration process operates at the individual level, with each agent having a unique ID.

Rather than optimizing a single global parameter set, each Agent ID acts as an index that links individual trajectory data to the corresponding parameter search space during the PSO process. This design allows the calibration algorithm to infer distinct behavioral characteristics, such as preferred speed range, acceleration and deceleration tendencies, and directional adjustment sensitivity, for every agent independently. The surrogate model utilizes the Agent ID embedding as part of its input feature space, ensuring that predictions are conditioned not only on environmental and motion-related variables but also on agent-specific patterns learned from the data. Consequently, the optimization process can adaptively tune parameters for each agent according to their personal dynamics and interaction style, yielding a population of agents with diverse behavioral profiles that better reflect the heterogeneity observed in real pedestrian movement.

3.1.3 Model Execution

This module integrates the preprocessed simulation inputs, applies the calibrated parameters, and executes the calibrated agent-based model (ABM). The goal is to reproduce pedestrian movement dynamics in a controlled virtual environment using calibrated agent configurations.

The execution process consists of two stages:

(1) Simulation Initialization

At the beginning of each simulation, agents are instantiated with calibrated behavioral parameters and placed within the reconstructed environment. The initialization step includes the following: At the beginning of each simulation, agents are instantiated with calibrated behavioral parameters and placed within the reconstructed environment. The initialization step includes:

- Number of agents: Total number of agents participating in the simulation episode.
- Environment spatial limits: The minimum and maximum (x,y) coordinates defining the simulation domain.
- Environment boundaries: A set of wall line segments annotated with labels and endpoint coordinates used for collision handling and distance computation.
- Pillars and obstacles: Circular obstacles defined by center coordinates and radius values, forming non-passable zones.
- Agent static properties: Each agent is assigned a unique ID and both global and agent-specific (calibrated) reference parameters, including body diameter, initial position, preferred speed factors, and sensitivity to environmental conditions.
- Indicated routes: A sequence of intended movement points (or guiding segments) extracted from real trajectories and used to approximate route-following behavior.

These components together define the full state of the simulation before any movement occurs.

(2) Velocity-Based ABM Execution

After initialization, the simulation proceeds in discrete time steps. At each step, a velocity-based movement model updates every agent's position and orientation while enforcing behavioral and physical constraints. For the update of each agent, this model incorporates the calibrated parameters, local environmental context, and interactions with surrounding agents.

Velocity-Based Movement Model

The velocity-based movement model is the core of the simulator, governing how agents perceive the environment, react to local conditions, and update their motion. The model integrates both individual behavior and environmental structure, and is parameterized by values optimized during calibration.

The main components are as follows:

- **Pedestrian Agent:** Each pedestrian acts as an autonomous decision-making entity. Using its calibrated parameters, it determines its next velocity based on current speed, desired direction, environmental cues, and interaction forces. Agents also retain short-term memory aspects such as previous velocity and recent turning behavior.
- **Agent Interactions:** Local interactions influence motion based on nearby agents' positions and velocities. The model incorporates:
 - proximity-based speed adjustments,
 - turning constraints when another agent is predicted to cause a collision, and
 - behavior modulation using condition codes (obstacle proximity and crowd density).

These interactions help capture realistic flow patterns such as lane formation, avoidance maneuvers, and speed modulation in dense regions.

- **Environment Representation:** The simulated environment is a 2D Euclidean space bounded by wall segments and enriched with structural elements such as pillars, designated passable nodes, and exit points. Agents continuously evaluate:
 - distance to the nearest wall or pillar,
 - motion angle relative to the to the boundary,

- pre-defined indicated route, and
- whether movement adjustments are required (e.g., slowdown near walls, detour around pillars).

These environmental cues feed directly into the velocity update mechanism.

Together, the above components form a complete movement model capable of reproducing both micro-level interactions and the macro-level pedestrian flow behaviors observed in real-world trajectory data.

Explanation of the Algorithm

The Algorithm 2 simulates pedestrian movement within a dynamic and spatially constrained environment, activating only a limited number of *scene agents* at a time, while, still reflecting realistic space utilization. The following outlines the main components of the algorithm:

Inputs

- **Environment setup** $\mathcal{E}$: Defines the spatial environment, including boundaries, obstacles (e.g., walls, pillars), and designated walking areas where pedestrians interact.
- **Agent pool** $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$: Represents all pedestrians activated for simulation. Each agent is initialized with personal attributes such as height, body size, initial position and velocity, route indication, and locally calibrated parameters.
- **Maximum scene size** N_{scene} : Determines the upper limit on the number of agents simultaneously active in the simulation. New agents are introduced dynamically when space becomes available.
- **Calibrated parameters** $\Theta_{\text{calibrated}}$: A collection of agent-specific parameter values derived from prior calibration (e.g., PSO), governing each agent's speed, angle adjustment, and interaction behavior.
- **Maximum simulation steps** T_{max} : Specifies the total duration or stopping condition for the simulation.

Outputs

- **State set** $\mathcal{S}$: The recorded simulation data containing each agent's position (x_i, y_i) ,

Algorithm 2: Scene-Limited Agent-Based Pedestrian Simulation

Input: All agents $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$, Environment $\mathcal{E}$, Calibrated parameters $\Theta_{\text{calibrated}}$, Global defaults Ω_{global} , Maximum steps T_{max} , Maximum scene size N_{scene}

Output: Recorded state set $\mathcal{S}$ across simulation steps

Initialize:

- Define environment $\mathcal{E}$ (walls, pillars, boundaries)
- Create agent pool $\mathcal{A}_{\text{pool}}$ (all agents prepared but not yet scheduled)
- Initialize active set $\mathcal{A}_{\text{scene}} \leftarrow \emptyset$
- Set time $t \leftarrow 0$, and initialize state log $\mathcal{S}$

while $t < T_{\text{max}}$ *and not all agents have completed* **do**

Dynamic Scene Scheduling:

while $|\mathcal{A}_{\text{scene}}| < N_{\text{scene}}$ *and unscheduled agents remain* **do**

Select next agent a_i from $\mathcal{A}_{\text{pool}}$

Add a_i to $\mathcal{A}_{\text{scene}}$

Initialize a_i state using $\Theta_{\text{calibrated}}$

end

Step() for all active agents:

foreach agent $a_i \in \mathcal{A}_{\text{scene}}$ **do**

IdentifyLocalConditions(a_i):

Detect walls, pillars, and neighboring agents affecting a_i

Compute local state $\mathcal{L}_i$ (distance codes, angles, interaction fields)

Select calibrated parameters θ_i conditioned on $\mathcal{L}_i$

or global defaults Ω_i if no matching local condition exists

Advance($a_i, \theta_i, \mathcal{L}_i$):

Adjust velocity and orientation based on selected rules

Update position:

$(x_i(t+1), y_i(t+1)) \leftarrow (x_i(t), y_i(t)) + v_i \cdot \text{dir}(\theta_i)$

Update internal motion state (heading, turning, collision response)

Record updated state of a_i into $\mathcal{S}$

end

Agent Completion Check:

Remove any agents that reached goal or exited environment from $\mathcal{A}_{\text{scene}}$

Increment time step: $t \leftarrow t + 1$

end

return Recorded state set $\mathcal{S}$

velocity v_i , orientation θ_i , and relevant local condition codes (e.g., obstacle and crowd proximity) at every time step.

Key Features of the Algorithm

- **Scene-Limited Scheduling:** Only a fixed number of agents (N_{scene}) are active within the simulation at any given time. As agents complete their routes or exit the scene, new agents are drawn from the pool, allowing continuous inflow and outflow dynamics that resemble real pedestrian environments.
- **Local Parameter Adaptation:** Each active agent adapts its movement to the localized conditions ($\mathcal{L}_i$), such as the presences of nearby obstacles, walls, or surrounding pedestrians, using either scene-specific parameters or globally calibrated defaults.
- **Dynamic Obstacle and Neighbor Avoidance:** Agents adjust their orientation and velocity in response to spatial constraints, preventing collisions with static objects and other pedestrians while maintaining realistic motion continuity.
- **Adaptive and Scalable Design:** The scene-limited approach allows the simulation to scale to large populations by focusing computational effort on agents currently influencing the environment. This improves both performance and realism, enabling context-dependent interactions without simulating all agents simultaneously.

3.1.4 Quality Assessment

The final module evaluates the simulation by comparing simulated and observed trajectories. While pedestrian simulation models can be assessed using a variety of evaluation approaches—including aggregate flow statistics, fundamental diagrams, density–speed relationships, or qualitative visual inspection—such methods often fail to capture trajectory-level discrepancies and individual behavioral fidelity. In contrast, similarity-based evaluation directly compares simulated and real pedestrian trajectories, enabling a fine-grained assessment of both spatial and temporal correspondence at the individual level.

Accordingly, similarity is quantified using distance-based measures such as Dynamic Time Warping (DTW), Fréchet distance, and Edit Distance on Real Sequences (EDR). These metrics capture how closely two sequences of positions or movements correspond in both space and time, providing a comprehensive assessment of the simulation’s ability to reproduce real-world pedestrian dynamics.

To perform this evaluation, trajectory similarity techniques were applied that explicitly account for spatial and temporal alignment, allowing for variations in speed, duration, and sampling rates.

Error reduction was computed at the agent level, where each agent’s calibrated simulation was evaluated against its corresponding baseline simulation using trajectory similarity metrics. This allows calibration effectiveness to be analyzed as a distribution across agents rather than a single aggregated value, thereby revealing heterogeneous calibration performance across individuals.

In the following, we first present the evaluation algorithm along with its explanation, and then describe the similarity measures employed in this study.

Similarity-Based Evaluation Algorithm

The following Algorithm 3 formalizes the procedure for evaluating simulated pedestrian states against real-world observations. The algorithm quantifies the similarity between simulated and observed sequences by using multiple distance metrics, thereby providing a rigorous measure of the model’s fidelity. Each of these captures complementary aspects of the trajectories, including

temporal alignment, spatial proximity, and robustness to noise or outliers.

Algorithm 3: Agent-wise Evaluation with Per-Agent Error Reduction

Input: Real trajectories $\mathcal{S}_{\text{real}} = \{S_{\text{real}}^{(i)}\}_{i=1}^N$,
 Baseline simulated trajectories $\mathcal{S}_{\text{base}} = \{S_{\text{base}}^{(i)}\}_{i=1}^N$,
 Calibrated simulated trajectories $\mathcal{S}_{\text{calib}} = \{S_{\text{calib}}^{(i)}\}_{i=1}^N$

Output:
 Agent-wise distance metrics for Baseline and Calibrated simulations,
 Agent-wise error reduction percentages

Preprocessing:
foreach agent $i = 1$ to N **do**
 Temporally align $S_{\text{real}}^{(i)}$, $S_{\text{base}}^{(i)}$, and $S_{\text{calib}}^{(i)}$
end
 Initialize dictionaries:
 $\text{metrics_base} \leftarrow \{\}, \text{metrics_calib} \leftarrow \{\}, \text{error_reduction} \leftarrow \{\}.$

Step 1: Compute agent-wise metrics for Baseline simulation
foreach agent $i = 1$ to N **do**
 Compute $\text{DTW}(S_{\text{real}}^{(i)}, S_{\text{base}}^{(i)})$
 Compute $\text{Fréchet}(S_{\text{real}}^{(i)}, S_{\text{base}}^{(i)})$
 Compute $\text{EDR}(S_{\text{real}}^{(i)}, S_{\text{base}}^{(i)})$
 Store results in $\text{metrics_base}[i]$
end

Step 2: Compute agent-wise metrics for Calibrated simulation
foreach agent $i = 1$ to N **do**
 Compute $\text{DTW}(S_{\text{real}}^{(i)}, S_{\text{calib}}^{(i)})$
 Compute $\text{Fréchet}(S_{\text{real}}^{(i)}, S_{\text{calib}}^{(i)})$
 Compute $\text{EDR}(S_{\text{real}}^{(i)}, S_{\text{calib}}^{(i)})$
 Store results in $\text{metrics_calib}[i]$
end

Step 3: Compute agent-wise error reduction
foreach agent $i = 1$ to N **do**
 foreach metric $\in \{\text{DTW}, \text{Fréchet}, \text{EDR}\}$ **do**
 $E_{\text{base}}^{(i)} \leftarrow \text{metrics_base}[i][\text{metric}]$
 $E_{\text{calib}}^{(i)} \leftarrow \text{metrics_calib}[i][\text{metric}]$
 Compute agent-wise error reduction:

$$r^{(i)} = \frac{E_{\text{base}}^{(i)} - E_{\text{calib}}^{(i)}}{E_{\text{base}}^{(i)}} \times 100\%$$

 Store $r^{(i)}$ in $\text{error_reduction}[i][\text{metric}]$
 end
end

Return:
 $\text{metrics_base}, \text{metrics_calib}, \text{error_reduction}$

Algorithm 3 is integrated into our modular framework and provides a flexible, extensible mechanism for both per-agent and per-feature assessment, accommodating single-agent cases

as well as large-scale multi-agent simulations. By systematically applying the selected similarity metrics, the framework can efficiently evaluate the fidelity of simulated pedestrian trajectories and thereby ensure their realistic correspondence to real-world observations.

Selected Similarity Measures

This subsection outlines the similarity measures used to evaluate the correspondence between simulated and real pedestrian trajectories including:

- **Dynamic Time Warping (DTW):** DTW measures the optimal alignment between two time-dependent sequences $X = (x_1, x_2, \dots, x_n)$ and $Y = (y_1, y_2, \dots, y_m)$ by minimizing the cumulative distance under temporal warping:

$$DTW(X, Y) = \min_{w \in W} \sum_{(i,j) \in w} d(x_i, y_j) \quad (3.5)$$

where W is the set of all possible warping paths and $d(x_i, y_j)$ is typically the Euclidean distance between two points. This measures comparison between trajectories of different lengths or speeds by stretching or compressing the time axis.

- **Fréchet Distance:** Fréchet distance captures the minimal “effort” needed to traverse both trajectories while preserving the temporal order of points. It can be intuitively described as the shortest leash length connecting two moving entities walking along the curves. For continuous curves $f, g : [0, 1] \rightarrow \mathbb{R}^2$, Fréchet distance is defined as follows:

$$d_F(f, g) = \inf_{\alpha, \beta} \max_{t \in [0, 1]} \|f(\alpha(t)) - g(\beta(t))\| \quad (3.6)$$

where α and β are continuous, non-decreasing reparameterizations of time. This metric emphasizes geometric shape similarity and is sensitive to the spatial structures of trajectories.

- **Edit Distance on Real Sequences (EDR):** EDR measures the minimum number of edit operations required to transform one trajectory into another, allowing tolerance for small spatial deviations. Given a threshold ϵ , the recursive formulation is:

$$EDR(i, j) = \begin{cases} 0, & i = 0, j = 0, \\ i, & j = 0, \\ j, & i = 0, \\ EDR(i - 1, j - 1), & \text{if } \|x_i - y_j\| < \epsilon, \\ EDR(i - 1, j), & \\ 1 + \min \left\{ EDR(i, j - 1), \right. & \text{otherwise.} \\ \left. EDR(i - 1, j - 1) \right\} & \end{cases} \quad (3.7)$$

Unlike DTW and Fréchet distance, EDR is robust to noise and local outliers, being focused on sequence similarity rather than strict geometric matching.

Collectively, these similarity-based metrics provide complementary insights: DTW emphasizes temporal alignment, Fréchet distance captures geometric conformity, and EDR quantifies sequence-level correspondence. Together, they ensure a holistic evaluation of spatial-temporal realism in simulated pedestrian trajectories.

3.2 Dataset

In this study, we used a real-world pedestrian trajectory dataset [52] collected in the indoor public environment of the Asian Trade Center (ATC) in Osaka, Japan. The dataset was recorded between October 24, 2012 and November 29, 2013, typically on Wednesdays and Sundays, and comprises a total of 92 observation days.

Each CSV file corresponds to one observation day, and each row represents a tracked individual at a specific timestamp. The dataset includes the following fields:

- *time [ms]* (*Unix time + milliseconds/1000*) — precise timestamp of the measurement, enabling reconstruction of continuous trajectories
- *person ID* — unique identifier assigned to each pedestrian to track movement over time
- *position X [mm]* — horizontal coordinate within the planar walking space

- *position Y [mm]* — vertical coordinate within the planar walking space
- *position Z (height) [mm]* — estimated height of the pedestrian, allowing analysis of body-size heterogeneity
- *velocity [mm/s]* — instantaneous walking speed computed at each timestamp
- *angle of motion [rad]* — direction of movement derived from velocity vectors
- *facing angle [rad]* — orientation of the pedestrian’s body, which may differ from the motion angle and is useful for interpreting turning behavior and head orientation

Table 3.2: Summary of dataset statistics and environmental information

Item	Value	Description
Environment Size	~900 m ²	Main hall in the west with multiple entrances and exits, including stairs, offices, restaurants, and a corridor in the east with shops and exit points.
Number of Unique Surrounding Spots	63	Distinct spatial reference points (spots) defined by line segments within the environment.
Number of Pillars	7	Structural pillars located within the environment.
Average Number of Agents per Day	38,895	Mean number of unique pedestrians observed per day.
Min Records per Agent	1	Minimum number of recorded data points for a single agent.
Max Records per Agent	600,072	Maximum number of recorded data points for a single agent.
Average Records per Agent	1,112.6	Mean number of trajectory points collected per agent across the dataset.
Average Number of Records per Day	39,508,124	Mean total number of trajectory records per observation day.
Recording Frequency	0.048 s	Average interval between consecutive recordings for each agent.
Average Covering Duration per Day	11.02 h	Total temporal span covered per day across all agents.

See Appendix A 6.2 for detailed per-day summaries of the full dataset.

Table 3.2 presents an overview of the spatial properties of the environment and the scale of the pedestrian trajectory dataset used in this study. The observation area is approximately 900m² and consists of a complex indoor layout with multiple entrances, exits, and functional zones, such as restaurants, offices, shops, and connecting corridors. Within this space, 63 distinct surrounding spots and 7 structural pillars are defined as reference elements for spatial analysis.

The dataset captures large-scale pedestrian activity, with an average of 38,895 unique agents recorded per day. Individual trajectories vary widely in length, from a single to over 600,000 records for a single pedestrian, but average 1,112.6 records per agent. This means that daily more than 39 million trajectory records are captured, at an average sampling interval of 0.048 s.

Each observation day comprises approximately 11 hours of continuous coverage, which enables detailed analysis of pedestrian movement patterns at both the individual and collective levels.

Table 3.3: Comparison of dataset statistics between Wednesday and Sunday recordings.

Weekday	Average Records	Average Agents	Max Records Per Agent	Average Records Per Agent	Recording Frequency (s)
Wednesday	28,891,780.48	22,561.20	600,072	1,300.65	0.0396
Sunday	50,124,468.37	55,228.96	358,293	924.55	0.0560

As shown in Table 3.3, in the dataset, a clear distinction appears between the recordings taken on Wednesdays and those taken on Sunday, of which there are 46-days each. Sunday sessions generated nearly twice the number of records and agents compared to Wednesdays, indicating higher pedestrian density during weekend observations. However, the average number of records per agent is higher on Wednesdays, suggesting longer continuous trajectories per individual—possibly reflecting regular commuters or workers with extended stay durations within the environment. The recording frequency also differs slightly, with Wednesday data captured at a higher temporal resolution (0.0396 s compared to 0.0560 s). Occasionally, national holidays falling on Wednesdays exhibit larger record volumes similar to weekends.

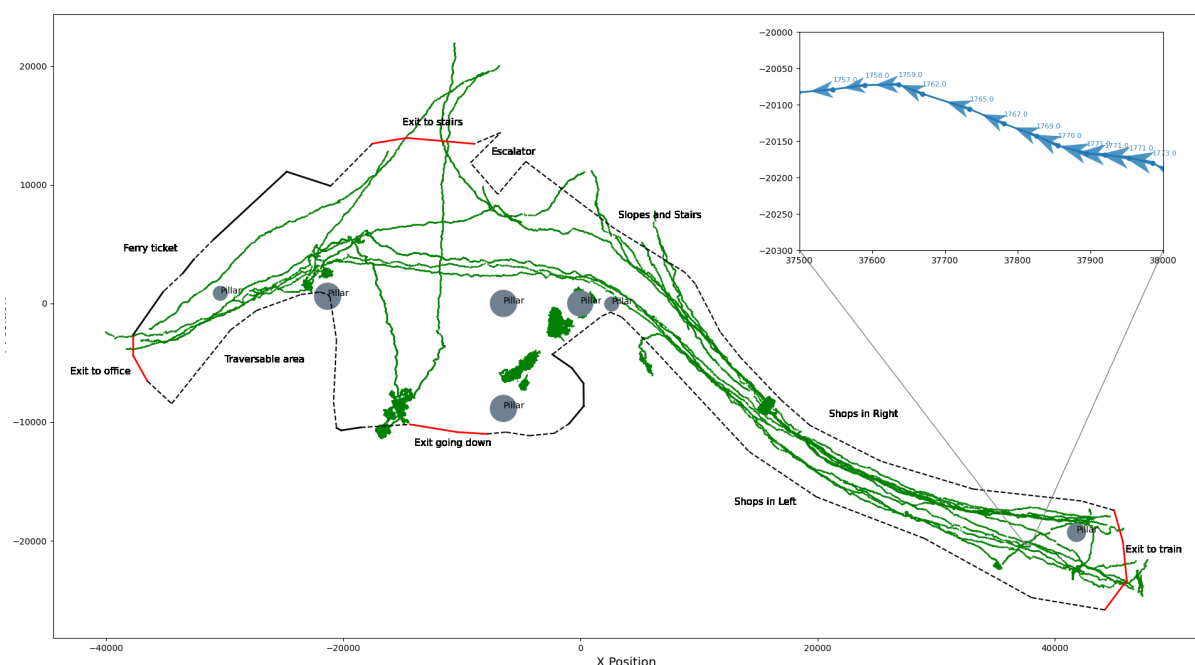


Figure 3.3: Overview of the ATC environment layout, showing spatial boundaries, pillars, and exits.

Figure 3.3 visualizes a sample of 50 pedestrian trajectory data points, shown in green.

Pillars are depicted as gray circles, while surrounding spots—including shops, stairs, slopes, and exits—are represented by line segments. Wall segments indicate non-accessible boundaries, whereas the remaining segments correspond to accessible areas within the environment. The main exits located on the west, east, north, and south sides are highlighted with red line segments to emphasize the primary pedestrian flow channels. The enlarged view in the upper-right corner illustrates that the dataset includes per-step motion angles as well as approximated pedestrian heights (Z-direction).

3.3 Particle Swarm Optimization

This study utilizes Particle Swarm Optimization (PSO) as the core method for calibrating agent-based pedestrian models. Inspired by the collective dynamics of social organisms, PSO is a population-based metaheuristic that efficiently explores complex, high-dimensional search spaces in Eberhart et al. [53] and Shi et al. [54]. Its gradient-free nature makes it especially well suited for closed-box simulations and non-differentiable objective functions, which are common in agent-based modeling.

Recent studies have demonstrated the effectiveness of PSO for tuning agent-level behavioral parameters. For example, Michels et al. [38] and Du et al. [39] successfully applied PSO to calibrate pedestrian and crowd simulation models, highlighting its ability to balance computational efficiency with robust optimization performance. Building on these insights, our work adopts PSO to systematically search for parameter sets that minimize discrepancies between simulated and real pedestrian trajectories.

3.3.1 Overview of PSO

Each particle in the swarm represents a potential parameter set and navigates the solution space according to both its personal best and the global best positions discovered so far. The procedure can be summarized as follows:

1. **Initialization:** A swarm of particles is initialized with random positions and velocities within the search space.

2. **Fitness Evaluation:** Each particle's fitness is evaluated using an objective function, which, in this study, quantifies the similarity between simulated and real pedestrian trajectories.
3. **Personal Best Update:** Each particle records its best position (*personal best*) based on its fitness value.
4. **Global Best Update:** The swarm identifies the best overall position (*global best*) among all particles.
5. **Velocity Update:** The velocity of each particle is updated as

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot (p_i - x_i^t) + c_2 \cdot r_2 \cdot (g - x_i^t), \quad (3.8)$$

where

- v_i^t is the velocity of particle i at iteration t ,
- x_i^t is its current position,
- p_i is its personal best position,
- g is the global best position,
- w is the inertia weight balancing exploration and exploitation,
- c_1 and c_2 are cognitive and social acceleration coefficients, and
- $r_1, r_2 \in [0, 1]$ are uniformly distributed random numbers.

6. **Position Update:** Each particle moves to a new position according to

$$x_i^{t+1} = x_i^t + v_i^{t+1}. \quad (3.9)$$

7. **Iteration:** Steps 2–6 are repeated until a termination condition is satisfied, such as reaching the maximum number of iterations or convergence to within a predefined threshold.

Through this iterative process, particles balance exploration (broad search) and exploitation (local refinement), enabling convergence toward high-quality solutions. For the present study,

fitness was evaluated based on the trajectory similarity between simulated and observed pedestrians, ensuring that the calibrated parameters effectively capture heterogeneous behavioral patterns.

3.3.2 Baseline Calibration using Standard PSO

To establish a baseline for model calibration, we employ the standard PSO algorithm, where each particle represents a candidate parameter set for all agents simultaneously. The fitness function minimizes the discrepancy between simulated and observed pedestrian states (e.g., velocity, movement-related parameters) across agents and over time.

This setup performs calibration as a bulk optimization process, while still allowing agent-level heterogeneity because the parameter vector inside each particle contains the individual parameters for every agent. However, this design introduces substantial computational cost, as the full simulation must be rerun at every PSO iteration to evaluate new candidate solutions. The complete procedure for standard PSO-based calibration is provided in Appendix B 6.2.

Challenges and Computational Efficiency

As the number of agents and parameters increases, the search space expands dramatically, leading to high computational demand. In homogeneous models, calibration involves keeping the set of global parameters shared across agents small in order to maintain the manageability of the optimization space. In contrast, heterogeneous calibration requires agent-specific parameter tuning, expanding the search space to $O(M \times K \times P)$, where M is the number of agents, K is the number of parameters per agent, and P is the number of particles.

This growth in dimensionality often necessitates larger swarm sizes to maintain adequate exploration and avoid premature convergence. To address this challenge, prior studies have suggested alternative strategies, such as Gaussian process models, Bayesian optimization, or surrogate-assisted optimization, to reduce the number of expensive simulation calls. To handle our large-scale trajectory dataset, we employ a deep learning-based surrogate model, which is particularly effective when trained on high-volume data.

The surrogate model is integrated directly into the PSO loop, which enables the model to

efficiently approximate simulation-based fitness evaluations and substantially reduces computation time while preserving calibration accuracy. The next section introduces our proposed deep learning–assisted heterogeneous calibration framework, which embeds this pre-trained neural surrogate model in order to accelerate optimization in high-dimensional spaces.

Chapter 4

Implementation and Results

4.1 Data and Surrogate Model Preparation

This section outlines the procedures for preparing the dataset and training the surrogate model. It explains how pedestrian trajectory data were selected and processed, how key environmental and behavioral features were extracted, and how the surrogate model architecture was chosen. The training and validation procedures to ensure the model’s reliability in reproducing pedestrian movement under various local conditions are also described.

4.1.1 Data Collection and Preprocessing

As described in Section 3.2, the complete dataset consists of 92 days of pedestrian trajectory observations. Clear differences are observable between weekdays and weekends, reflecting distinct mobility patterns: weekday data generally capture regular commuter flows, whereas weekend data tend to exhibit higher pedestrian volumes and more diverse movement behaviors.

For the experiments, a single representative dataset was selected from a **weekend day** (June 23, 2013, Sunday). This date was chosen because it exhibits the highest average number of trajectory records per agent among all observation days, making it particularly suitable for evaluating complex, high-density pedestrian scenarios.

It is important to note that the framework itself does not depend on a specific date. The only requirement is that the input consist of timestamp-ordered trajectories with consistent person

identifiers. Therefore, any dataset satisfying these conditions can be readily integrated into the proposed pipeline.

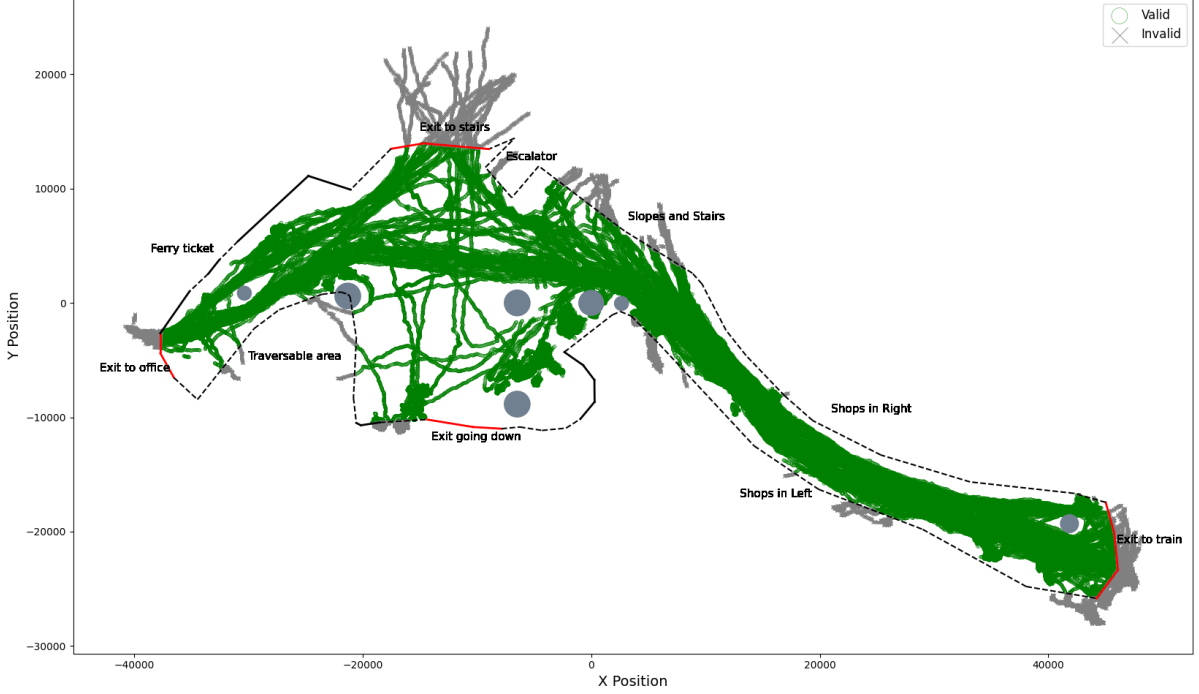


Figure 4.1: Green circle markers (O) indicate valid trajectory points used in the experiment, while gray cross markers (X) represent out-of-bound points excluded.

Figure 4.1 illustrates the trajectory points used in the experiments, which were extracted as subsets from the cleaned dataset by randomly selecting a specified number of agents (100, 200, 500, 800, or 1,000) after shuffling the full agent list to avoid any ordering bias.

Table 4.1: Summary of trajectory record statistics for selected agent subsets.

Number of Agents	Minimum Records	Maximum Records	Mean Records
100	14,159	179,601	24,612.37
200	9,777	179,601	18,074.45
500	6,289	179,601	11,838.97
800	5,048	179,601	9,493.10
1000	4,542	179,601	8,549.46

minimum threshold of 4,000 records per agent was applied.

Table 4.1 summarizes the minimum, maximum, and mean number of records per agent. A minimum threshold of 4,000 data points per agent was enforced based on exploratory analysis, resulting in a total of 1,295 agents satisfying this criterion.

The resulting dataset was subsequently divided into calibration, validation, and surrogate-model training/testing subsets, as described in Subsection 3.1.1.

4.1.2 Variable Experimental Instances

As outlined in Section 3.1.1, the deep-learning surrogate model is constructed using features corresponding to **Local Conditions**, **Reference Parameters**, and **Tunable Variables** to ensure consistency with the overall framework.

The *Local Conditions* component captures environmental variability through encoded condition codes that represent factors such as proximity to walls, distance to pillars, and surrounding crowd density. The *design parameters*, including view angle, crowd radius, and distance thresholds to walls and pillars, directly influence these features and therefore affect how the surrogate model perceives the local spatial context.

In the computational experiments, the *Reference* and *Tunable* parameters were held constant to ensure a fair comparison across runs, while the *Local Condition Codes* and the associated design parameters were systematically varied to examine their influence on model performance.

A number of variables, design choices, and hyperparameters in the proposed framework can be modified across experimental settings. These include the following:

- **Data Pipeline Module:** minimum number of data points per agent, number of local condition types, level of detail per condition, design parameters (e.g., view angle, proximity scaling factors), and surrogate model training parameters (e.g., maximum number of epochs, learning rate, network depth).
- **Model Calibration Module:** maximum number of PSO iterations, PSO hyperparameters (c_1, c_2, w, k, p), number of particles, and number of tunable parameters per agent.
- **Model Execution Module:** number of simulation steps

Based on preliminary experiments, several hyperparameters, such as minimum number of data points per agent and number of simulation steps, were fixed in order to maintain comparability. The remaining parameters, particularly the design variables and PSO settings, were treated as experimental factors because of their sensitivity and impact on calibration

quality.

While the general feature structure remains consistent, the level of the corresponding features are instantiated differently across experimental scenarios. Each experimental instance varies in terms of **Local Condition Codes**, **design parameter values** (e.g., view angle, crowd radius, distance thresholds), and **PSO variables** (e.g., number of iterations, number of particles), enabling systematic exploration of environmental and behavioral variations.

For example, some experiments restrict wall and pillar codes to three levels (0 = neutral, 1 = relaxing, 2 = approaching) under fixed design parameters, whereas others employ more granular codes (0 = neutral, 1 = more relaxed, 2 = relaxed, 3 = approaching, 4 = touched, 5 = collided). Likewise, design parameters such as view angle, crowd radius, maximum number of PSO iterations, and number of particles are varied within predefined ranges in order to capture a broader spectrum of plausible pedestrian behaviors.

However, as discussed in Section 3.3.2, increasing the number of local condition codes leads to a proportional, or often even an exponential, growth of the parameter space. This substantially increases computational cost during model calibration, particularly when using optimization-based methods such as Particle Swarm Optimization. Thus, identifying a practical balance between computational efficiency and calibrated model quality is a central challenge. Achieving effective management of this trade-off is crucial for enabling exploration of diverse environmental configurations and agent behaviors while maintaining the tractability of the framework for large-scale simulations and repeated experimental runs.

Table 4.2: Illustrative examples of experimental instance structures.

Instance ID	View Angle	Crowd Radius	Wall & Pillar Threshold	Wall Code	Pillar Code	Crowd Code	PSO Iter.	Particles
1	$[-30^\circ, 30^\circ]$	$\sim$ height	velocity	0,1,2	0,1,2	0,1,2	50	10
2	$[-45^\circ, 45^\circ]$	$\sim$ height $\times 2$		velocity $\times 2$	0-4	0-4	0-4	10
3	$[-60^\circ, 60^\circ]$	$\sim$ height $\times 1.5$	velocity $\times 2$	0-2	0-4	0-4	100	30
N	...	...	...	...	...	...	...	...

Table 4.2 presents representative experimental instances generated from different combinations of local condition codes, design parameter values, and PSO calibration settings. Each instance includes not only environmental and behavioral configuration parameters (e.g., view angle, proximity thresholds, encoded local conditions) but also the optimization-related settings used during calibration, namely the maximum number of PSO iterations, number of particles,

total calibration runtime, and the resulting error reduction percentage.

This organization illustrates that the surrogate model’s feature space remains structurally consistent across all experiments, while variations are introduced through different instantiations of local condition codes, design parameters, and PSO settings.

Since the number of agents was not considered a critical factor for determining suitable parameter values for the design and optimization variables (although it naturally affects runtime for model training and calibration), we first conducted a comprehensive set of 48 experiments using 100 agents. These experiments systematically explored the full range of variable combinations to identify well-performing parameter configurations. For these 48 instances, the Crowd Radius was set proportional to the agent’s *height*, the Wall & Pillar Threshold was set proportional to the agent’s *velocity*, the Pillar Code was treated as a single option set $0,1,2$, and the number of local condition records was fixed at 200. The full sensitivity analysis, including experiments with other parameter combinations, is presented in Section 4.4.

In the results, *Instance 24* achieved the best overall performance, so its configuration was subsequently reused for experiments with larger agent populations (200, 500, 800, and 1,000). In addition, baseline experiments were conducted for five different agent counts to enable consistent comparative evaluation under the same configuration.

Detailed explanations of the experimental setup are provided in Appendix C 6.2. The experimental results in later sections will be compared against the outcomes of these representative instances.

Surrogate Model Implementation and Feature Engineering

The proposed deep learning surrogate model uses a consistent set of features that represent three main categories: **Local Conditions**, **Reference Parameters**, and **Tunable Variables**. These features capture the agent’s local environment, prior motion, and controllable behavioral parameters, and serve as the input for the model regardless of specific experimental settings.

Local Conditions are encoded as categorical features to represent environmental contexts and agent interactions, such as proximity to walls, pillars, and surrounding pedestrians. **Reference Parameters** are numerical features capturing the agent’s previous velocity and angles

relative to nearby structures, while **Tunable Variables** represent controllable behavioral adjustments, such as speed and angle change percentages.

The general feature set used across all experiments is summarized in Table 4.3.

Table 4.3: General set of input features and output label used in the surrogate model.

Inputs	Feature Name	Description
Local Condition (Categorical)	agentID	Encoded identifier of the current agent.
Local Condition (Categorical)	wallCode	Encoded direction of the nearest wall relative to the agent.
Local Condition (Categorical)	pillarCode	Encoded direction of the nearest pillar relative to the agent.
Local Condition (Categorical)	crowdCode	Discretized proximity code representing surrounding crowd density, calculated relative to the agent's body size.
Reference (Numerical)	relativeAngleWall	Angle between the agent's heading and the nearest wall.
Reference (Numerical)	relativeAnglePillar	Angle between the agent's heading and the nearest pillar.
Reference (Numerical)	prevSpeed	Agent's speed at the previous time step.
Tunable (Numerical)	speedUpPercentage	Percentage increase in speed at the current step.
Tunable (Numerical)	speedDownPercentage	Percentage decrease in speed at the current step.
Tunable (Numerical)	angleChangePercentage	Percentage of turning angle adjustment at the current step.
Tunable (Numerical)	surrSensitivity	Sensitivity factor to surrounding agent conditions.
Output (Label)	velocity	Predicted agent velocity for the current step.

Surrogate Model Architectural Selection and Training Accuracy

To tune the speed adjustment parameters based on stepwise agent conditions, we required a model capable of capturing complex relationships between input features (e.g., position, velocity, surroundings) and output behaviors. We selected TabNet [55], a deep learning architecture designed for structured data, which combines high interpretability with strong performance in sequential decision tasks.

TabNet's attention mechanism dynamically focuses on relevant features at each step, making it ideal for modeling agent behavior under changing conditions. Its interpretability also provides insights into the key factors influencing agent movement, which supports further model refinement

Training and Validation Accuracy

We conducted experiments on model training using varying numbers of epochs and learning rates. Significant improvements in model performance were observed as the number of epochs increased, with stabilization occurring after early stopping criteria were met.

Model performance was evaluated using *RMSE*. For the scenario with 100 agents, training loss decreased by approximately 88.21% and validation RMSE improved by about 67.3% up to the best epoch, demonstrating substantial convergence. Based on these results, we selected a learning rate of 0.001 and trained the model while applying early stopping with *patient* = 20 to prevent overfitting. After training, the model was exported for later use in the parameter tuning task.

Table 4.4: Training and validation RMSE reduction for different numbers of agents. Early stopping indicates the best-performing epoch.

Number of Agents	Training / Test Data	Early Stopped Epoch	RMSE Reduction (%)	
			Training	Validation
100	42,772 / 17,111	132	88.21	67.34
200	81,059 / 32,427	108	86.14	60.01
500	189,749 / 75,932	121	90.12	65.36
800	267,463 / 107,036	141	88.56	62.26
1000	301,646 / 120,691	126	89.65	69.63

As shown in Table 4.4, model training for each agent count stopped at the reported epoch due to early stopping triggered by overfitting. The number of training and test samples is determined per local condition using the MAX_RECORDS) criterion described in Section 3.1.1. Note that the samples closest to the mean value of each local condition were selected to ensure more accurate training.

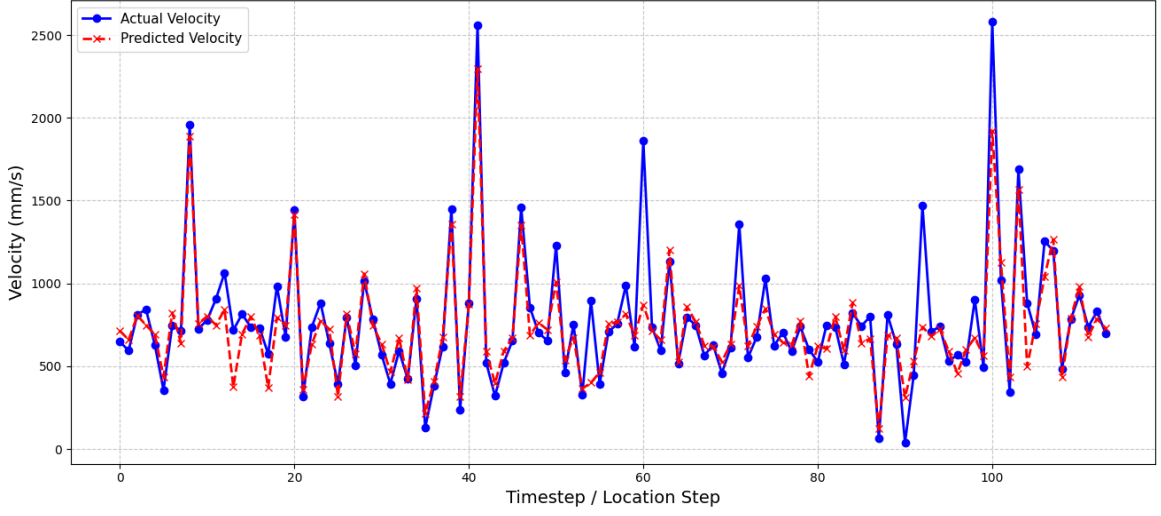


Figure 4.2: Predicted vs. actual velocity.

Figure 4.2 shows a sample prediction result for the next-step velocity, based on the current step information, including local condition features and reference values. As illustrated in the graph, the predicted velocities align closely with the actual values, suggesting that the model is capable of generalizing well despite partial data imputation.

4.2 Experimental Setup and Environment

4.2.1 Computational Environment

The computational environment in the present study included a high-performance workstation with an Intel(R) Core(TM) i9-13900 (32 CPUs) processor, 64 GB RAM, and a 32GB NVIDIA GeForce RTX 4070 Ti GPU.

The proposed framework utilizes the following software libraries and frameworks:

- *Core Implementation:* Python (version 3.x).
- *Deep Learning Model:* TabNet implementation in PyTorch-4.1, trained with a learning rate of 0.001 and a maximum of 1,000 epochs. Early stopping with a patience of 20 epochs is applied to prevent overfitting.
- *Parameter Tuning:* Particle Swarm Optimization (PSO) implemented with *pyswarm*, $\text{num_iterations} \in \{10, 50, 100\}$, and $\text{num_particles} \in \{1, 5, 10\}$ based on experi-

mental results evaluating convergence, with the following hyperparameters: 'c1': 0.5, 'c2': 0.3, 'w': 0.9, 'k': 3, 'p': 2.

- **Movement Model:** Velocity-based movement model implemented using the *Mesa* framework, utilizing *Mesa.Time* for simultaneous agent activation and *Mesa.Space* for continuous spatial representation.
- **Visualization:** The simulation results were visualized using the *matplotlib* library, which enabled high-quality, publication-ready plots. Additionally, the *PyQt* library was utilized to enable faster and more efficient visualization of the simulation process.

4.2.2 Simulation Environment and Input Integration

The simulation environment was constructed using information from ATC Team [52], which details the camera system and its mapping to real-world coordinates. This layout incorporates the locations and types of the surrounding walls, pillars (defined by their radii and center positions), and designated exit points.

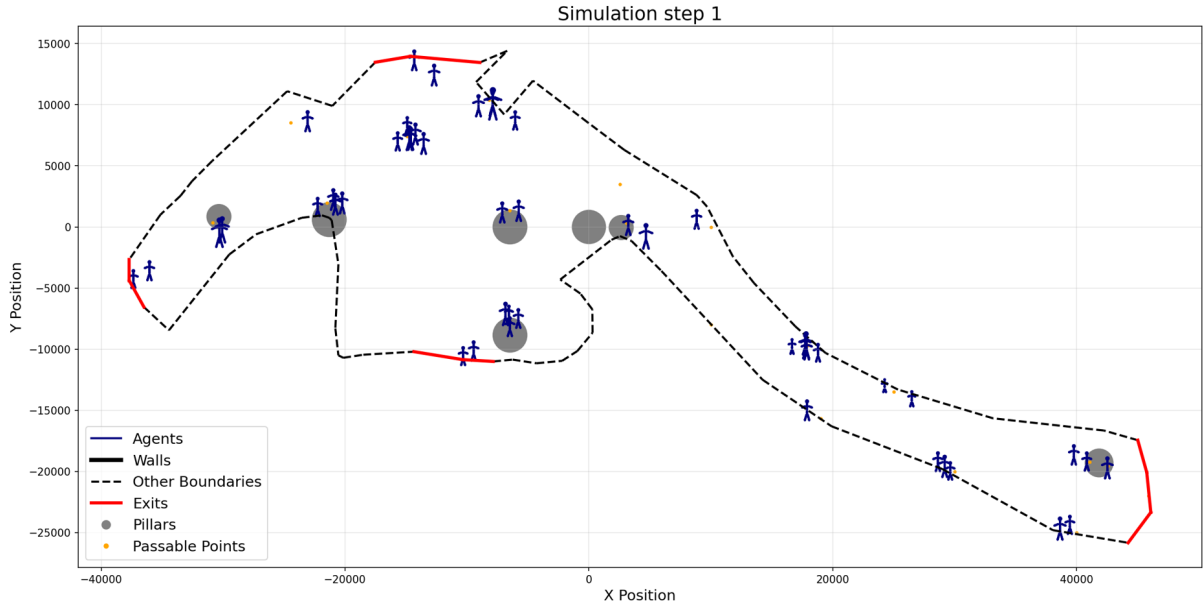


Figure 4.3: All agents are initialized around the own starting point.

Figure 4.3 illustrates the initialization of the simulation, where each agent is assigned the following categories of properties:

- static attributes, such as ID, height, and body diameter;
- an initial position, randomly selected from one of the indicated route points;
- a tailored set of local and global parameters for movement; and
- an indicated route that includes specific waypoints to reach.

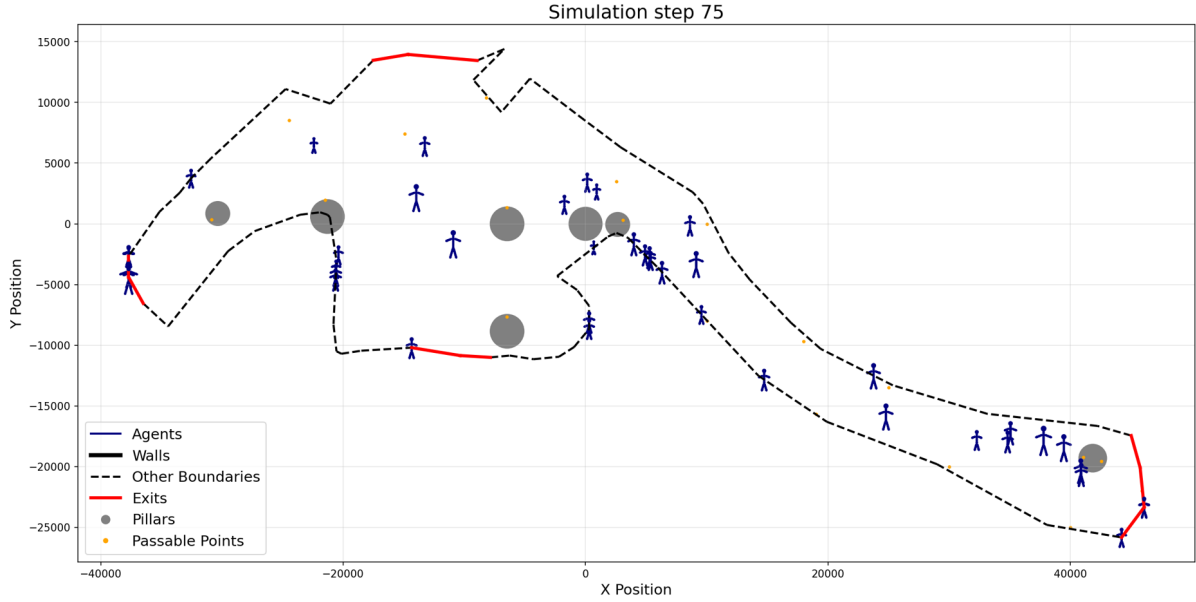


Figure 4.4: Agents are moved through the environment at step 75

As shown in Figure 4.4, all agents aim to reach their targets by navigating through a sequence of passable points while avoiding collisions with walls, pillars, and other agents while resolving potential deadlocks.

Agent interactions are governed by a two-phase movement model of “step” and “advance” that incorporates collision avoidance, movement dynamics, speed regulation, and angle adjustments based on calibrated parameters.

For visualization purposes, agent sizes are scaled up beyond their actual dimensions, which may result in visual overlap, particularly in dense areas.

4.3 Main Results: Model Calibration and Validation

This section presents the primary findings of our study, including outcomes from model calibration, validation performance, and additional insights.

To evaluate the effectiveness of the calibration process, we analyzed calibration accuracy across various agent group sizes: 100, 200, 500, 800, and 1000 agents. We compared the performance of the deep learning–assisted PSO with the standard PSO, using experimental instances corresponding to Instance IDs 24 and 49–57.

4.3.1 Model Calibration Performance

For each configuration, calibration was performed over 100 iterations with 10 particles in the Model calibration module. The fitness function was defined using a Dynamic Time Warping (DTW) similarity metric, which quantified the correspondence between real and predicted velocity sequences under the respective local conditions for each agent.

Calibration Runtime

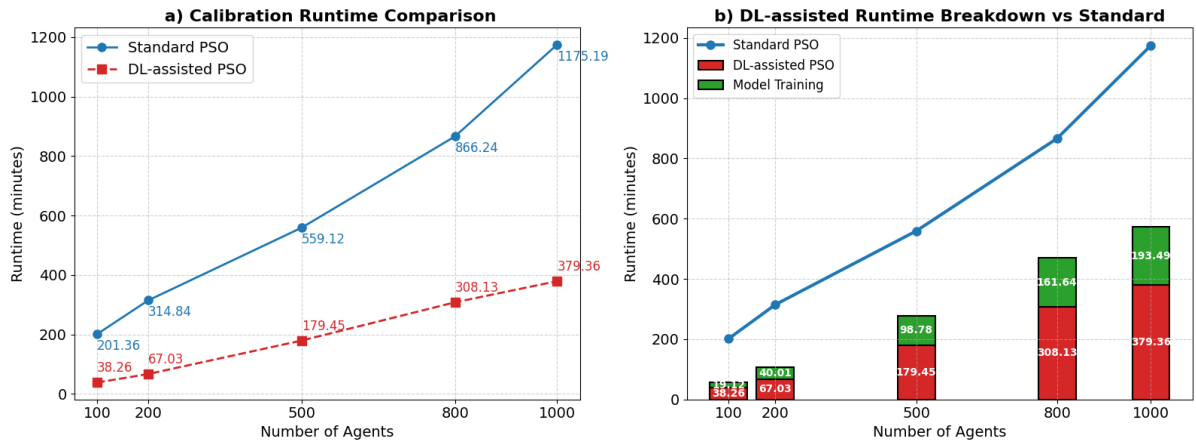


Figure 4.5: Calibration runtime comparison.

Figure 4.5 presents a calibration runtime comparison between the standard PSO and the deep learning–assisted PSO for different numbers of agents, ranging from 100 to 1,000 to reflect small- to large-scale simulation scenarios.

In the results, deep learning–assisted PSO consistently outperformed the standard PSO in terms of runtime. For instance, with 100 agents, the standard PSO required approximately 201.36 minutes, whereas the deep learning–assisted PSO only needed 38.26 minutes, a reduction of over 80%.

The results also demonstrated that the runtimes for both methods increase with the number of agents, reflecting the expected growth in computational demand for larger simulations. However, the deep learning–assisted PSO scales more efficiently, have shown a sub-linear increase relative to the standard PSO. This improvement is primarily due to the model’s ability to provide better initial guesses for the PSO search, thereby reducing the number of iterations and evaluations required to achieve a comparable calibration accuracy.

Overall, the figure demonstrates that the Deep-Learning-assisted PSO not only reduces runtime compared to the standard method but also maintains its advantage when the model training overhead is considered, highlighting its suitability for large-scale agent-based simulation calibration.

Calibration Accuracy

Calibration accuracy for each agent was assessed by comparing the initial fitness function value (prior to calibration) with the best fitness value achieved after PSO convergence. For each agent group size, we recorded the best, worst, and mean calibration accuracy across all agents.

Table 4.5: Comparison of calibration accuracy between standard PSO and deep learning–assisted PSO across different agent counts.

Number of Agents	Standard PSO			Deep-Learning-assisted PSO		
	Best (%)	Worst (%)	Mean (%)	Best (%)	Worst (%)	Mean (%)
100	25.12	18.31	21.71	59.12	50.12	54.23
200	26.78	17.03	19.94	57.91	48.36	52.61
500	26.05	14.61	18.12	60.31	49.15	53.48
800	25.17	13.64	20.56	58.68	48.98	53.08
1000	27.03	13.31	19.07	60.69	46.34	53.12

Table 4.5 presents the calibration results obtained using the standard PSO and the deep learning–assisted PSO, respectively. The standard PSO evaluates the entire simulation for all agents at each PSO step, relying solely on local environmental conditions and providing a general parameter tuning approach. In contrast, the deep learning–assisted method uses agent-specific information (e.g., agent ID), which enables individualized parameter optimization and behavioral heterogeneity among agents to be captured.

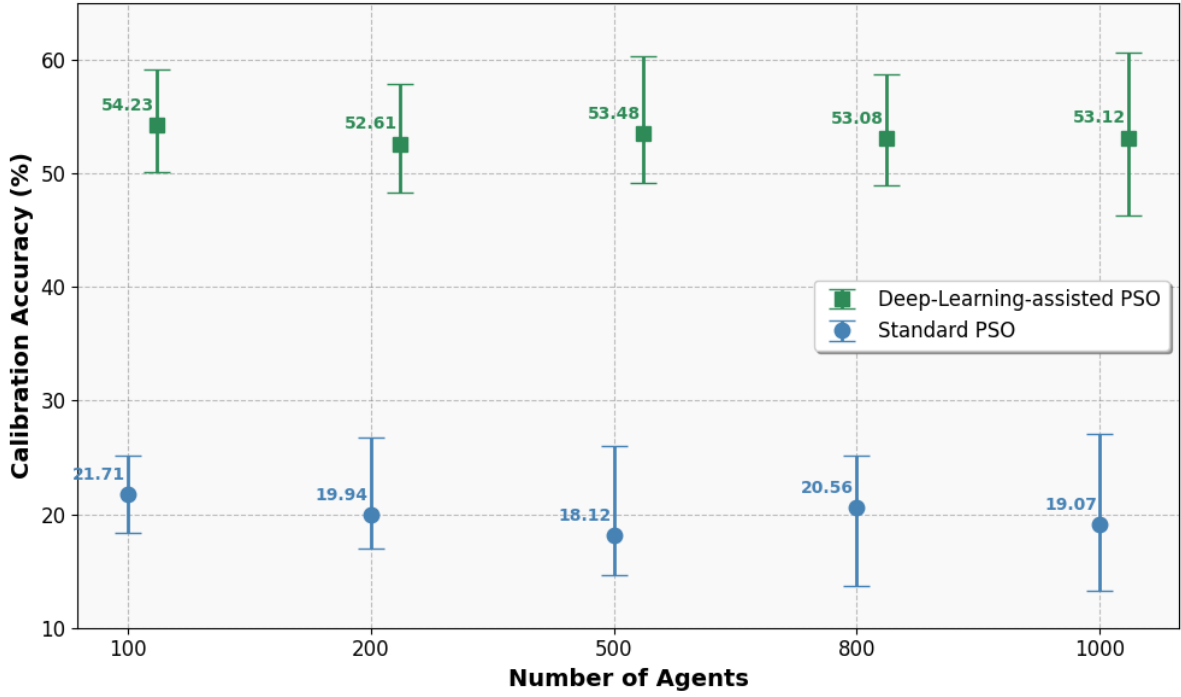


Figure 4.6: Calibration accuracy vs. number of agents , evaluated using DTW-based fitness.

Figure 4.6 illustrates the calibration accuracy trends for both the standard PSO and deep learning–assisted PSO for different numbers of agents (100, 200, 500, 800, and 1,000). For each agent group size, the mean calibration accuracy is shown along with error bars representing the range between the best- and worst-performing agents.

The results indicate that the deep learning–assisted PSO consistently achieved higher calibration accuracy than the standard PSO for all agent group sizes. For example, the range of the mean accuracy of the deep learning–assisted method was approximately between 52% and 54%, whereas the standard PSO achieved mean values between 18% and 22%. This demonstrates that incorporating agent-specific information allows for more precise parameter tuning and better captures behavioral heterogeneity among agents.

The error bars also illustrate that the variability among agents is reduced with the deep learning–assisted PSO, suggesting that the method provides more robust calibration to different agent behaviors. Overall, the figure draws attention to the computational and accuracy advantages of integrating deep learning and PSO, particularly for larger-scale simulations.

Effect of PSO Iterations on Agent-Level Calibration Accuracy

To evaluate the impact on model calibration of increasing the number of Particle Swarm Optimization (PSO) iterations, we analyzed agent-specific improvements in the fitness function for three number of iterations—10, 50, and 100—using $numParticles = 10$. Calibration accuracy for each agent was quantified as the percentage reduction in the fitness function from its initial value to the best value achieved during the PSO iterations. Lower fitness values indicate better alignment between model output and real data, reflecting more accurate parameter tuning.

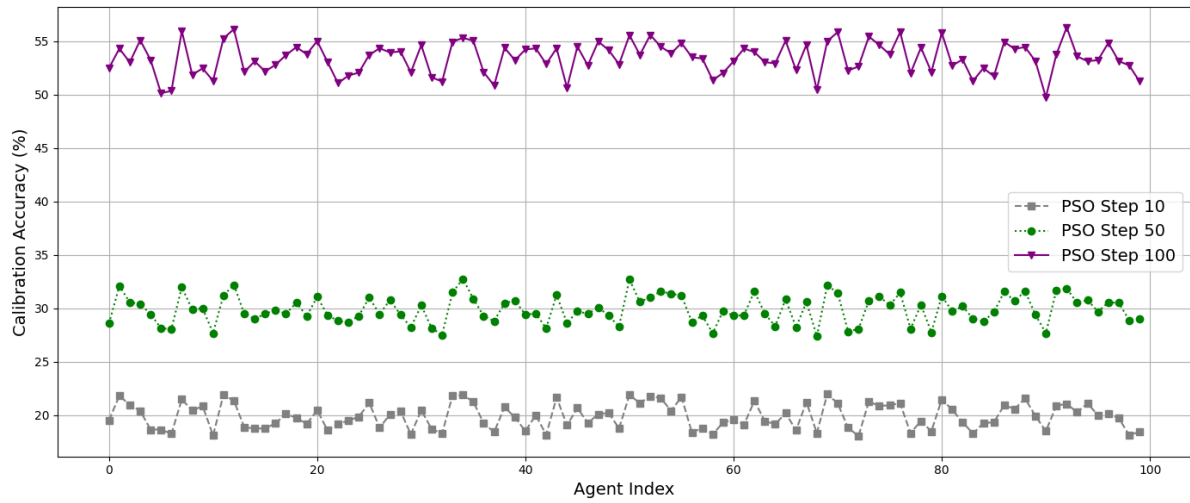


Figure 4.7: Calibration accuracy (fitness function reduction) vs. PSO iteration for 100 agents.

As shown in Figure 4.7, calibration accuracy improves consistently as the maximum number of PSO iterations increases. At 10 iterations, most agents exhibit modest improvements, achieving error reductions of approximately 18–23%. Increasing the iterations to 50 leads to more substantial gains, with error reductions in the range 23–32%, indicating progressively better parameter tuning. After 100 iterations, the majority of agents achieve error reductions exceeding 50%, though the improvements begin to plateau, suggesting convergence in calibration for most agents.

According to these findings, the variability across agents of calibration accuracy becomes more apparent with additional iterations, reflecting the differing complexity between agents of optimizing parameters. This fact demonstrates the strength of PSO in adaptively calibrating agent-specific parameters, which over time gives both more overall model robustness and greater per-agent accuracy.

4.3.2 Overall System Validation Results

Visual Inspection and Property Testing of Simulated Agents

Before conducting statistical validation, we performed a visual inspection to confirm that both the static and dynamic properties of the simulated agents behaved as expected. Each agent is initialized with predefined static parameters such as body diameter, initial speed, and starting position. As the simulation progresses, these agents update their dynamic states including location, orientation, and speed, in response to environmental factors and local interactions.

To validate these dynamics, we visualized the behavior of a representative agent throughout its trajectory. This includes the spatial path from the start to the target, orientation vectors indicating movement direction, speed variations over time, and obstacle avoidance behavior.

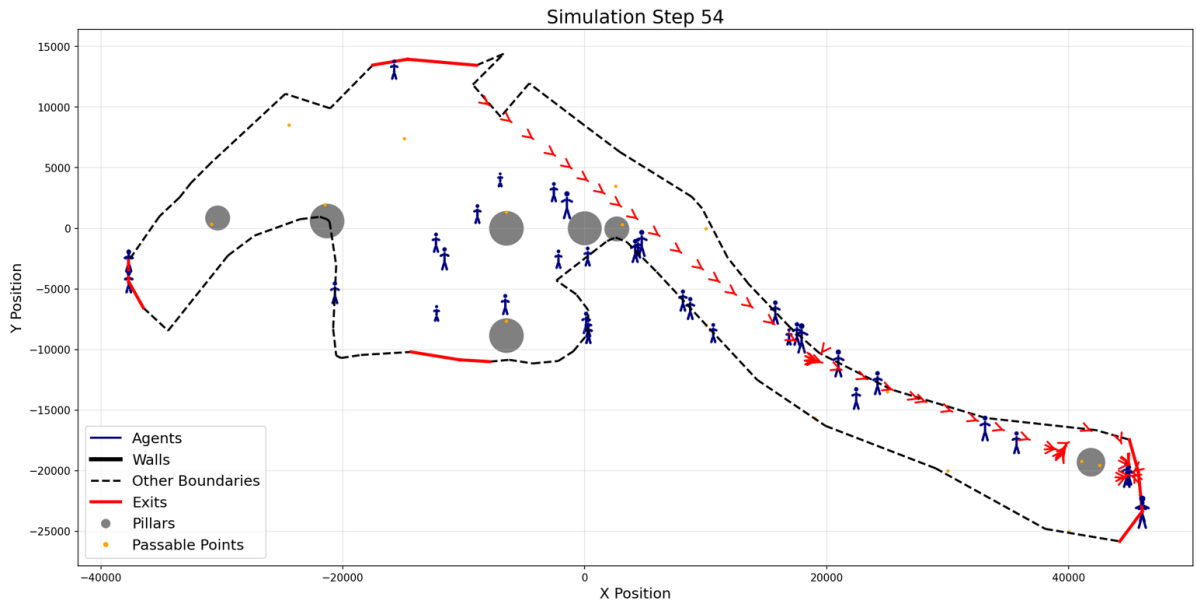


Figure 4.8: Visualization of a simulated agent's trajectory, orientation, and interaction dynamics.

The example in Figure 4.8 illustrates the dynamic behavior of a sample agent, showing the ability of the agent to adapt its movement in real time. The agent initialized near escalator (upper center), proceeds through multiple assigned targets, and actively avoids collisions along the way. After reaching one of the targets, it reverses its direction while avoiding collisions demonstrating a change in orientation for the subsequent step before continuing toward the next target.

This qualitative inspection provides initial confirmation that agent behaviors follow the expected physical and behavioral principles, serving as a foundation for subsequent quantitative evaluation.

Agent-Based Simulation Validation via Similarity-Based Evaluation

To evaluate the quality of the simulated trajectories, we adopted a systematic error assessment workflow that utilized multiple **similarity metrics** explained in in Section 3.1.4. These metrics consistently measured the closeness of the simulated agent trajectories to the observed real-world trajectories, regardless of the specific error type. This unified evaluation approach is applied to a suite of similarity metrics including DTW, Fréchet Distance, and Edit Distance on Real sequence (EDR) in order to provide a comprehensive assessment of simulation accuracy under the heterogeneous calibration strategy.

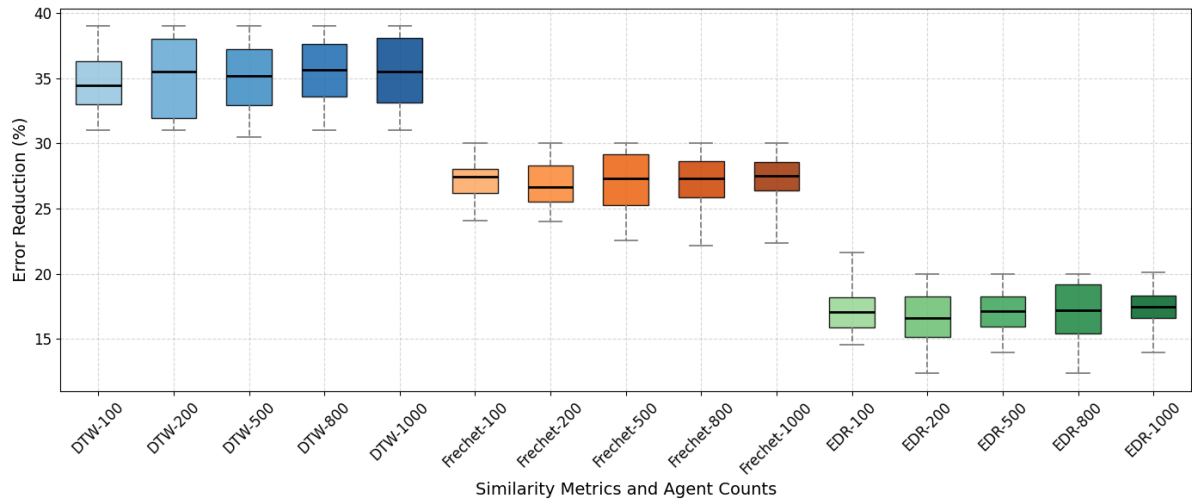


Figure 4.9: Validation accuracy for different metrics and agent counts

Figure 4.9 presents the distribution of validation performance across the DTW, Fréchet, and EDR similarity metrics for increasing agent counts from 100 to 1,000. Within each metric, the five box plots associated with the respective agent counts are arranged from left to right and are rendered using a color gradient that deepens progressively, making performance trends easier to visually compare.

Key observations:

- DTW consistently shows higher median error reduction values, indicating strong alignment with real trajectories across agent densities.
- Fréchet and EDR exhibit slightly lower median performance but provide complementary perspectives on trajectory similarity and model fidelity.
- As agent counts increase, variability (interquartile range) tends to increase slightly, reflecting greater heterogeneity in pedestrian interactions under denser conditions.

Overall, the Figure 4.9 demonstrates that the calibrated model maintains stable validation performance across multiple metrics and scales of pedestrian density, supporting the robustness of the simulation framework.

Detailed Validation Results for 500 Agents

To further examine the effectiveness of the proposed (deep learning–assisted PSO) calibration method, we conducted a detailed analysis using 500 agents and generated 500 sample evaluations per similarity metric. Figure 4.10 presents two complementary visualizations: (1) a combined boxplot–swarm plot showing the distribution of error-reduction values for DTW, Fréchet, and EDR metrics, and (2) overlapping histograms illustrating the density distribution for each metric.

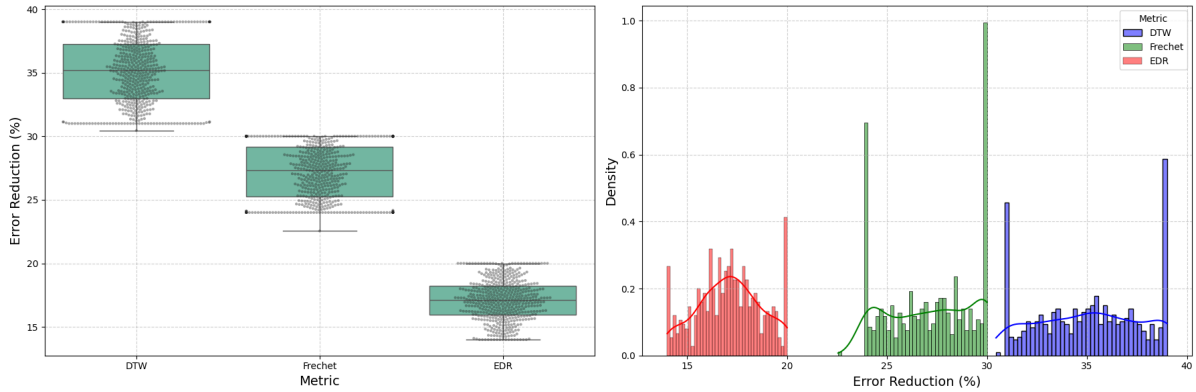


Figure 4.10: Error reduction and distribution for 3 similarity metrics (DTW, Fréchet, and EDR) with heterogeneous tuning using 500 agents and 500 samples.

The left panel of Figure 4.10 reveals the variability and central tendency of error reduction achieved by the heterogeneous method across the three similarity metrics. The boxplots show that DTW and Fréchet metrics exhibit consistently higher median error reduction values, indicating stronger improvements under heterogeneous tuning. The accompanying swarm plot

highlights agent-level spread and confirms that most agents cluster around higher improvement values, demonstrating robustness in the method.

The right panel shows the density distribution of error reduction values for each metric. DTW presents the sharpest and highest density peak, suggesting strong and consistent improvement across agents. Fréchet also shows substantial improvement with a relatively narrow distribution, while EDR exhibits a wider spread and a slightly lower central tendency. This broader variability in EDR arises partly from the metric’s internal matching process, which can generate more pronounced outliers and pull values farther apart—reflecting its lower sensitivity to local temporal variations compared with DTW and Fréchet.

Overall, these results demonstrate that the heterogeneous calibration approach offers statistically meaningful performance gains across all similarity metrics. It is especially effective for metrics sensitive to fine-grained temporal and geometric differences, such as DTW and Fréchet, where improvements are both larger and more consistent.

However, it is important to note that agents with **richer local condition data** (for example, those experiencing more diverse interactions or environmental variations) tend to achieve higher error reduction. Such agents have more informative behavioral signals for the learning-assisted calibration to exploit. Conversely, agents with sparse or less variable data may realize more modest gains, which reflects a natural limitation in reconstructing fine-grained real-world behavior when the available input information is limited.

Real vs Predicted speed comparison by Local Conditions

We further evaluated the overall system performance by comparing agent speeds, as our velocity-based movement model relies on finely tuned parameters to ensure accurate simulation. This analysis considered 200 agents sampled across all experiments, focusing specifically on neutral local conditions—cases where external social or environmental constraints (e.g., walls, obstacles, or surrounding agents) did not influence the agent’s movement. This filtering ensured that the simulated speeds represent natural walking behavior and could be fairly compared to the real-world data.

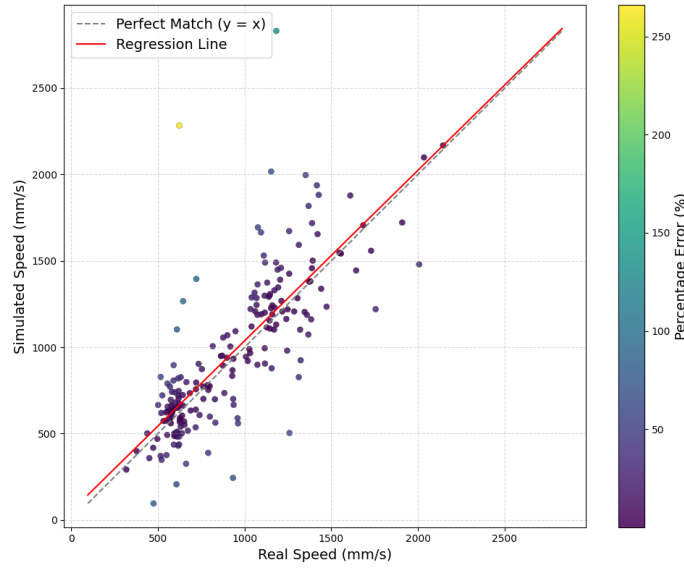


Figure 4.11: Scatter plot comparing real and simulated speeds for 200 agents.

Figure 4.11 presents the results as a scatter plot, where each point represents one instance of real (x-axis) and simulated (y-axis) speed under the same neutral conditions. The points are colored according to the percentage error relative to the real speed, allowing for a clear visual distinction between low-error and high-error predictions. The gray dashed identity line indicates perfect agreement between real and simulated speeds. A red regression line is also included to highlight the overall trend in the simulated outputs.

The regression line lies slightly above the identity line, indicating a systematic overestimation of walking speed in the simulated output. Despite this tendency, the overall accuracy is strong: 146 out of 200 samples (73%) had a percentage error of less than 30%, suggesting good agreement between real and simulated values under different local conditions. A small number of points with speeds exceeding 2000 mm/s (2 m/s) were observed, which likely correspond to agents who were running rather than walking.

Velocity-Based Spatial Utilization Analysis

This subsection presents a two-step spatial analysis framework for evaluating pedestrian movement behavior using velocity-based heatmaps. The analysis focuses on the corridor area of the environment, which is expected to exhibit higher pedestrian density and more frequent interactions due to its role as a primary circulation space. Such regions are therefore particularly

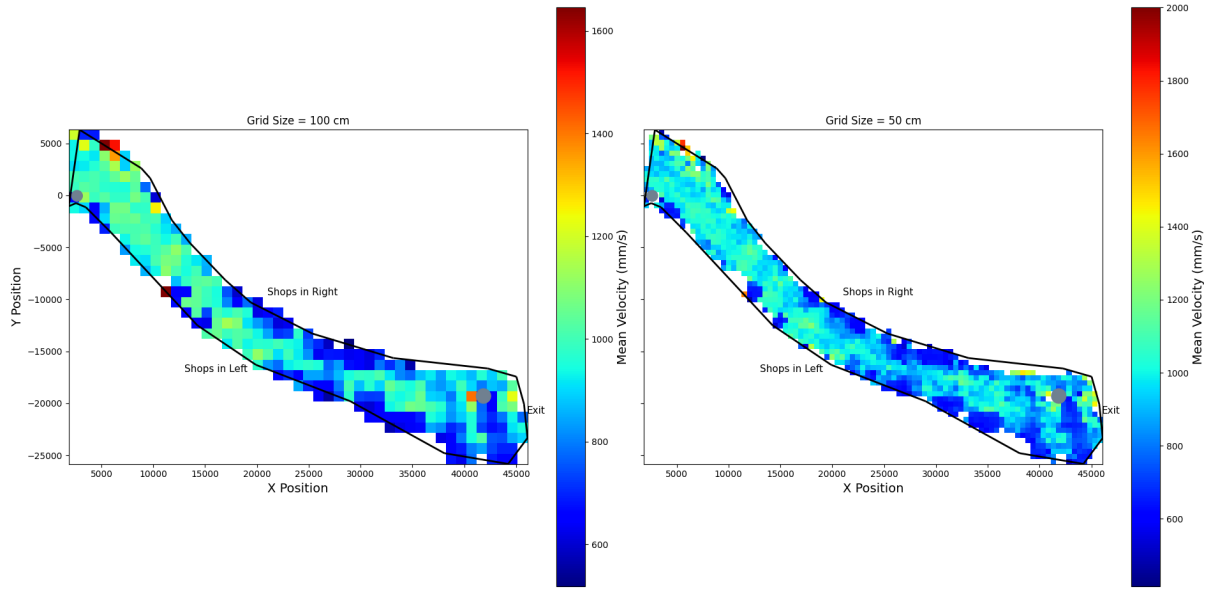


Figure 4.12: Velocity-based spatial utilization heatmaps of real pedestrian trajectories at two spatial resolutions (100 cm and 50 cm grids).

informative for understanding collective movement patterns and local behavioral adaptations.

First, we visualize and analyze empirical spatial utilization patterns derived from real pedestrian trajectory data. Second, we outline a spatial comparison approach for quantifying discrepancies between real and simulated velocity fields based on overlapping spatial regions, enabling a spatially consistent evaluation of simulation fidelity within the most behaviorally relevant areas of the environment.

(1) Visualization and Analysis of Real Velocity Heatmaps The spatial utilization of pedestrian movement is first examined using velocity-based heatmaps computed from real trajectory data. The accumulated dataset consists of approximately 1.2 million pedestrian trajectory records collected in the corridor environment. By discretizing the space into regular grid cells and calculating the mean walking velocity within each cell, we obtain a spatially continuous representation of pedestrian behavior.

Figure 4.12 illustrates the resulting velocity heatmaps at two spatial resolutions. The coarser grid captures global flow structures, while the finer grid reveals localized behavioral variations. Across both representations, higher mean velocities are consistently observed along the central corridor region, indicating relatively unobstructed pedestrian flow. In contrast, reduced velocities are evident near shop-front areas and corridor boundaries, where pedestrians

are more likely to slow down due to visual engagement, interactions, or localized stopping behavior. These empirical heatmaps provide a spatially grounded reference for subsequent simulation-based analysis.

At the coarser resolution (100 cm), the heatmap captures the global structure of pedestrian flow within the corridor. Higher mean velocities are predominantly observed along the central region of the corridor, indicating relatively unobstructed and goal-oriented movement. In contrast, lower velocities appear near shop-front areas and along the corridor boundaries, where pedestrians tend to slow down due to visual attention, localized interactions, or stopping behavior.

The finer-resolution heatmap (50 cm) reveals more detailed spatial heterogeneity that is not visible at the coarser scale. Localized low-velocity regions become more pronounced near shops, obstacles, and transition zones, while subtle velocity gradients across the corridor width can be clearly observed. These fine-scale patterns are partially smoothed out in the coarser representation, demonstrating the importance of spatial resolution in capturing micro-level pedestrian behavior.

(2) Absolute Difference Analysis Between Real and Simulated Heatmaps To quantitatively evaluate the spatial agreement between real and simulated pedestrian behaviors, an absolute difference heatmap is computed over the overlapping spatial regions of the two datasets. The comparison is conducted at a grid resolution of 50 cm, enabling a fine-grained assessment of local behavioral discrepancies.

Only grid cells that are occupied in both the real and simulated datasets are included in the analysis, ensuring a fair comparison that is not biased by differences in spatial coverage. For each overlapping grid cell, the absolute difference between the simulated and real mean velocities is calculated.

The simulated dataset consists of trajectories generated from a 500-agent simulation, comprising 317,691 records. Approximately 68% of the grid cells observed in the real data are overlapped by the simulated data and are therefore used in the comparison.

Figure 4.13 illustrates the resulting absolute difference heatmap, highlighting localized regions where the simulation either overestimates or underestimates pedestrian walking speeds

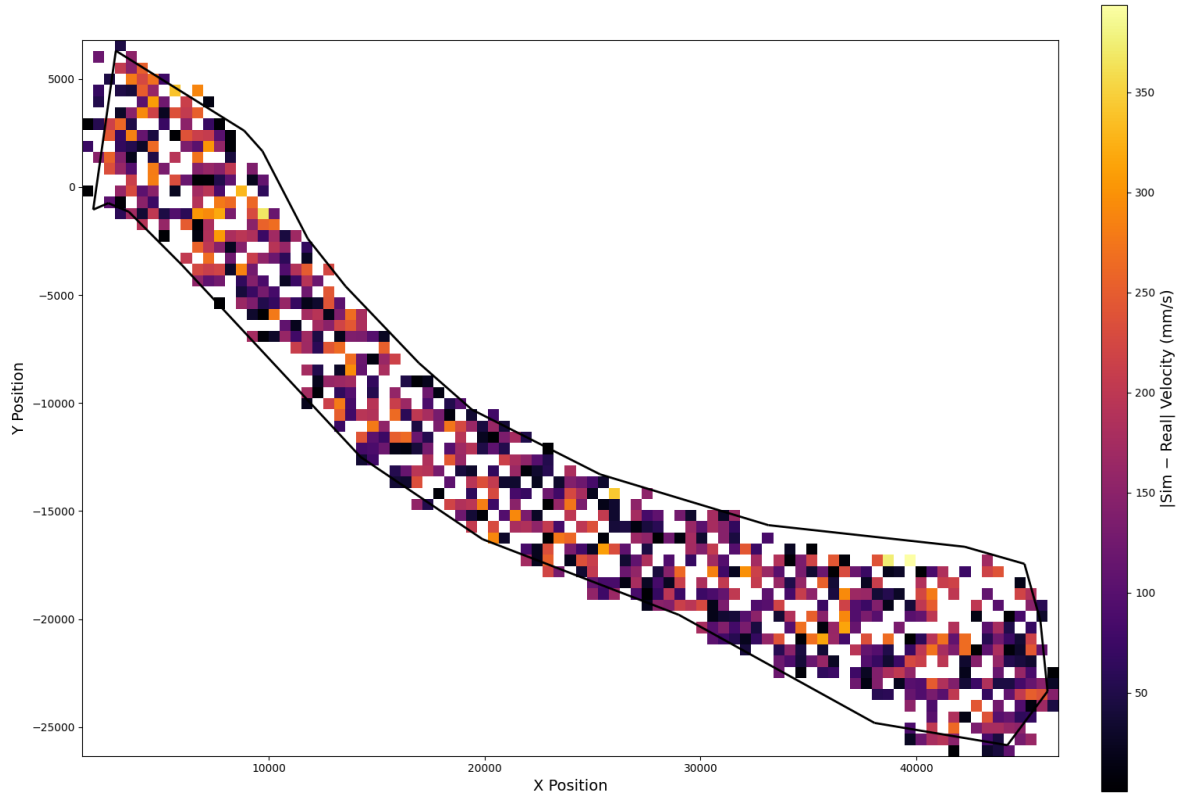


Figure 4.13: Absolute difference heatmap of mean pedestrian velocities between real and simulated data (50 cm grid).

relative to empirical observations. The maximum observed velocity difference is 29.53%, while the minimum difference is 2.31%.

Overall, the velocity-based spatial utilization analysis reveals consistent behavioral trends in both datasets. Pedestrians tend to increase their walking speed in the central corridor areas, where interactions and environmental constraints are limited, whereas reduced velocities are consistently observed near shop fronts, corridor boundaries, and other high-interaction regions. These empirically observed spatial patterns provide a critical behavioral reference for validating and refining simulated pedestrian dynamics.

4.4 Systematic Sensitivity and Experimental Design

To evaluate the robustness, reliability, and interpretability of the proposed framework, a systematic sensitivity analysis that quantifies how variations in design parameters and hyperparameters affect model performance and behavioral fidelity is conducted. Following design of experiments (DoE) principles, the analysis explores the parameter space efficiently and identifies the most

influential factors through an automated and reproducible experimental pipeline.

The analysis considers the following key categories of design parameters:

- **Data Pipeline:** Number of data points per agent, level of local condition detail, view angle, surrounding-agent approximation, and Deep Learning surrogate model settings.
- **Model Calibration:** PSO hyperparameters (inertia w , cognitive/social coefficients c_1 , and c_2), iteration count, particle count, and the number of agent-level parameters being optimized.
- **Model Execution:** Number of simulation steps, initial and adaptive speed settings, acceleration/deceleration percentages, and direction-related motion parameters.
- **Quality Assessment:** Factors influencing velocity deviation, trajectory similarity, spatial usage, and interaction density.

A structured three-stage sensitivity evaluation is performed:

1. **Local Sensitivity (OFAT):** Each parameter is varied independently to observe the direct effects on output metrics.
2. **Interaction Analysis (LHS):** Multiple parameters are varied simultaneously using Latin hypercube sampling to capture nonlinear interactions and combined effects.
3. **Global Sensitivity (Sobol):** Variance-based decomposition quantifies the contribution of each parameter and their higher-order interactions to overall output variability.

Sensitivity outcomes are assessed using several normalized metrics:

- **Velocity Deviation (Δv):** Difference between simulated and observed speed patterns.
- **Directional Consistency ($\Delta \theta$):** Angular deviation between simulated and real motion directions.
- **Spatial Utilization Ratio (SUR):** Comparison of space occupation and dispersion relative to real trajectories.
- **System Stability Index (SSI):** Variability across repeated simulations under identical settings.

These metrics allow consistent comparison across experimental configurations and help to identify both highly sensitive parameter regions and areas of stable model behavior.

4.4.1 Results and Discussion

We present and discuss sensitivity analysis results based on the experiments described in Appendix C 6.2.

A total of 48 experimental configurations were executed using 100 agents. Each configuration varied four primary factors: (i) view angle range ($[-30, 30]$ or $[-60, 60]$), (ii) wall-condition codes, (iii) crowd-condition codes, and (iv) PSO calibration parameters (numbers of iterations and particles). Calibration runtime and quality scores (best, worst, and mean) were recorded for each instance.

Overall, the results exhibit three clear patterns:

1. PSO parameters dominate both runtime and model quality. Increasing the number of iterations or particles consistently increases calibration runtime (from approximately 4–8 seconds to as long as 40–44 seconds) and improves quality scores. This confirms PSO hyperparameters as the most influential factors in the calibration process.

2. View angle range moderately affects quality. Configurations with a wider view angle ($[-60, 60]$) generally achieve better quality scores, indicating that agents with a broader field of perception produce smoother and more realistic behaviors.

3. Wall and crowd condition codes introduce nonlinear effects. Configurations with smaller condition sets (e.g., obstacle codes with three levels 0 = neutral, 1 = relaxing, 2 = approaching) show only slightly lower quality, but this effect becomes more pronounced when combined with wider view angles or more PSO iterations, suggesting interaction effects between perception and calibration variables.

Table 4.6: Detailed sensitivity analysis summary (100-agent configuration).

Parameter	Values	Mean Quality	Runtime Impact	Observed Trend / Interpretation
View Angle	$[-30, 30]$	27.9	Low	Narrow FOV reduces contextual information; quality decreases.
	$[-60, 60]$	26.6	Moderate	Wider FOV yields more stable and consistent predictions.
PSO Iterations	10	16.3	Very Low	Under-calibrated; insufficient search depth.
	50	24.1	Medium	Balanced search; captures major parameter interactions.
	100	49.0	High	Best calibration accuracy but highest computational cost.
Number of Particles	5	27.8	Low	Faster convergence but prone to local minima.
	10	29.1	Moderate	Larger swarm improves exploration and quality.
Wall Codes	$\{0,1,2,3\}$	27.8	High	More detailed environmental encoding improves realism and tends to increase the overall quality
	$\{0,1,2\}$	26.5	Moderate	Missing one wall level reduces behavior fidelity and the overall quality.
Crowd Codes	$\{0,1,2,3\}$	27.7	High	Full crowd state representation improves consistency.
	$\{0,1,2\}$	26.4	Moderate	Reduced state information leads to weaker predictions.

As shown in Table 4.6, the overall sensitivity ranking derived from these 48 experiments is as follows:

1. maximum number of PSO iterations (strongest influence)
2. Number of particles
3. View angle
4. Crowd and wall condition codes

These findings provide a basis for refining parameter search ranges and constitute the first step toward more comprehensive global sensitivity analyses such as Latin hypercube sampling and Sobol variance decomposition.

4.4.2 Trade-Off Analysis: Runtime vs Quality

To evaluate the trade-off between calibration runtime and quality, we analyzed all 58 calibration instances:

- Instances 1–48 correspond to the Deep-Learning-assisted PSO method for 100 agents
- Instances 49–52 correspond to the Deep-Learning-assisted PSO for other numbers of agents (200, 500, 800, and 1000)
- Instances 53–58 represent the baseline approach (Standard PSO) for agent counts of 100, 200, 500, 800, and 1000.

The analysis was performed using a Pareto front approach.

Figure 4.14 illustrates the relationship between calibration runtime and mean quality for each instance.

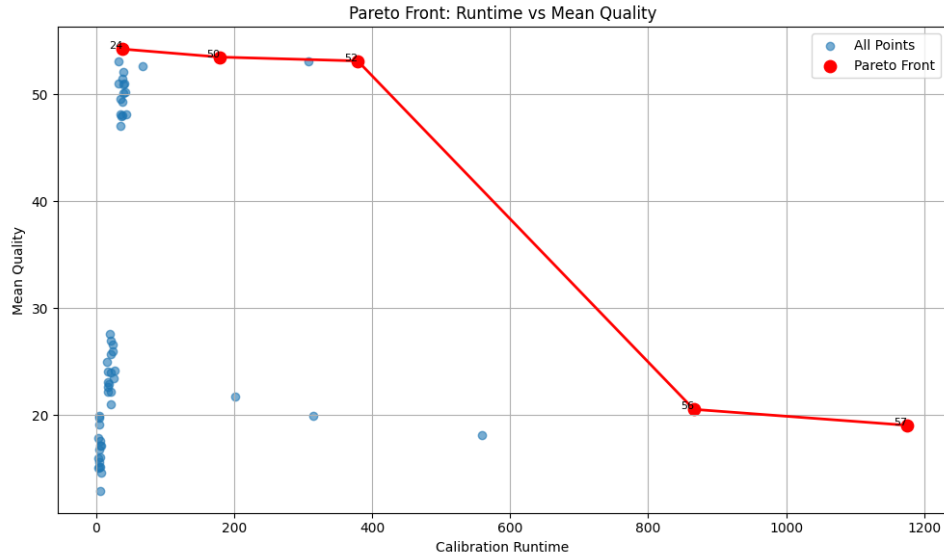


Figure 4.14: Pareto front of calibration runs.

As shown in Figure 4.14, each blue point corresponds to a calibration instance, and the red points represent the Pareto-optimal solutions. These are the instances for which no other calibration run achieves both a lower runtime and a better mean quality simultaneously. Notably, **Instance 24 achieved the best performance**, with a lower calibration runtime and the highest mean quality. Instances 50 and 52 also performed well, achieving high quality at a moderate runtime. Conversely, Instances 55 and 57 represent the worst cases, exhibiting the longest

runtime with relatively low quality. The figure highlights the set of optimal configurations that balance computational efficiency and calibration accuracy, providing insight into the most favorable calibration runs for practical deployment or further analysis.

Chapter 5

Discussion

5.1 Relating PSO-Tuned Parameters to Agent-Based Model Behavior

The Particle Swarm Optimization (PSO) calibration method implemented in this study was designed to identify the optimal parameter bounds governing each agent's movement and interaction behavior within the pedestrian agent-based model. The tunable parameters include the following:

- minimum and maximum walking speed,
- percentage changes associated with acceleration and deceleration,
- turn-angle change percentage, and
- sensitivity to nearby agents and local crowding conditions.

These parameters were calibrated with the objective of enabling the model to reproduce realistic pedestrian trajectories while maintaining behavioral plausibility and adhering to empirically observed ranges. Through PSO, agents were assigned individualized parameter sets reflecting heterogeneous movement characteristics, which improved the model's ability to capture diverse pedestrian behaviors.

A central issue emerges from this process concerning the interpretation of these PSO-tuned parameters in the context of the model's internal behavioral rules: while individual parameters

have intuitive interpretations (e.g., higher crowd sensitivity typically leads to slower movement in dense regions), their effects within the full simulation are not always linear or independent. For example, crowd sensitivity interacts with turn-angle flexibility, acceleration constraints, and initial state variables, potentially producing complex emergent outcomes. An agent with high sensitivity may not always slow down if its turning capacity allows it to maneuver away from obstacles efficiently; conversely, an agent with limited turning freedom may exhibit abrupt speed reductions even when sensitivity is moderate.

These coupled effects highlight the need for a deeper understanding of parameter interactions and their contributions to macroscopic simulation outcomes. The PSO procedure identifies combinations that improve fidelity to real trajectories, but it does not directly reveal why certain configurations perform better or how individual parameters influence emergent patterns.

To improve the interpretability and generalizability of the calibrated model, future research should extend this work by

- systematically analyzing multi-dimensional parameter interactions;
- conducting structured sensitivity analyses that vary parameters jointly and independently;
- evaluating how PSO-derived parameter sets behave under different environmental layouts, densities, and inflow conditions; and
- and identifying which parameters (or parameter clusters) most strongly influence behavioral realism.

Such analyses would help quantify how perturbations in speed bounds, turning behaviors, and crowd responses propagate through the simulation and shape emergent movement patterns. This, in turn, would improve model explainability, support transferability across environments, and provide clearer guidelines for applying PSO calibration to other pedestrian datasets or scenarios.

5.2 Extending the Framework Toward Single and Group Pedestrian Modeling

Individual-based modeling provides an effective foundation for understanding micro-level motion decisions, as it captures agents' responses to local environmental cues and neighbor interactions. While the current framework focuses on individual-level modeling and calibration, pedestrian behavior in real environments is often shaped by social interactions within groups. Groups of pedestrians, whether as pairs, families, or friends, frequently exhibit collective patterns that differ substantially from those of individuals, including spatial cohesion, mutual coordination, and adaptive responses to environmental constraints. Empirical observations indicate that group members adjust their speeds, directions, and inter-personal distances to preserve social comfort and visual communication. Recent studies, including our preliminary analysis (S.Amartaivan and H.Morita [56]), as well as other work on pedestrian group dynamics (Yi et al. [57] and Saeed, R. A. et al. [29]), have demonstrated that pedestrians moving as a group (group pedestrians) tend to occupy larger spatial regions, maintain intra-group formations, and display distinct space utilization patterns compared to single pedestrians moving independently. These behavioral couplings highlight the necessity of expanding the current modeling framework beyond purely individual representations, suggesting that incorporating group-level dynamics into an agent-based framework could significantly improve behavioral realism and predictive fidelity.

The current agent-based framework is well suited for extension toward group modeling because it already uses parameterized behavioral rules. Several additional parameters could be introduced to represent group dynamics:

- in-group cohesion strength,
- coordinated turning adjustment,
- group-level velocity adaptation, and
- preferred social spacing.

Such parameters could be calibrated using the same PSO-based procedure, allowing group-

level behavior to be optimized alongside individual behavior.

To support this extension, trajectory data would need to include group identifiers, enabling extraction of features such as intra-group distance, group spread, and directional consistency. The calibration process could then be adapted to consider both individual-level and group-level behavior, potentially through a hierarchical optimization structure.

5.2.1 Preliminary Observations and Analytical Potential

Preliminary analyses of real pedestrian trajectories suggest that group pedestrians differ from single pedestrians not only in speed and spacing but also in how they utilize the environment. Groups tend to occupy central corridors, reduce direct interactions with walls, and show higher synchronization of turning behavior.

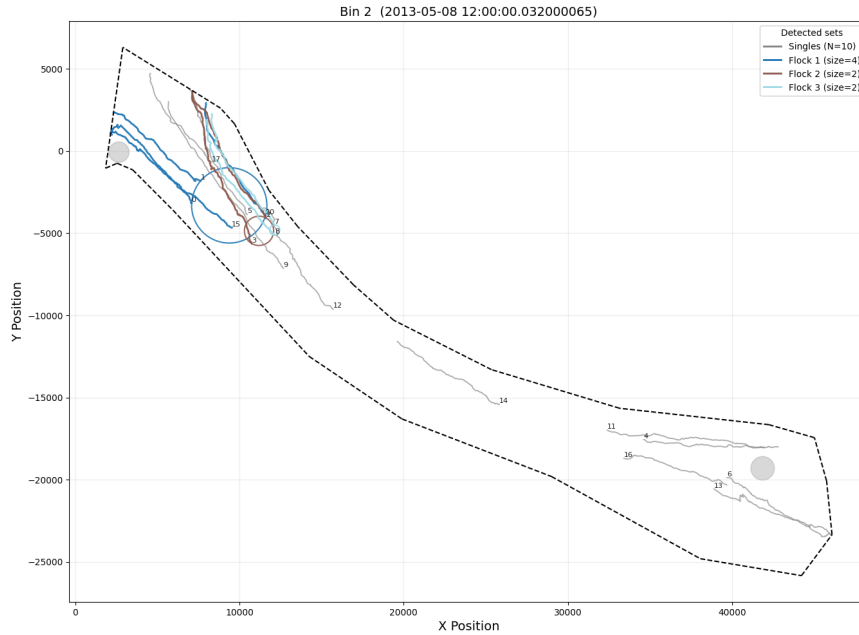


Figure 5.1: Sample flock classification results of pedestrian trajectories.

Figure 5.1 shows a visualization depicting 18 pedestrians within the observed scene previously discussed in the preliminary analysis by Amartaivan and Morita [56]. Among the pedestrians, 8 were identified as group pedestrians, and the remaining 10 were classified as single pedestrians. The spatial layout highlights how group members tend to remain in close proximity and collectively occupy a compact region of the environment, whereas single pedes-

trians are more dispersed. This visual distinction provides an intuitive understanding of group presence and spatial clustering, which later supports the quantitative analysis of group-induced space utilization patterns.

Quantitatively, these behaviors can be analyzed through space utilization heatmaps, interaction frequency distributions, and motion correlation metrics. The above discussion indicates that extending the current framework to include group-level behavioral modeling could reveal deeper insights into spatial organization and collective motion patterns.

To analyze pedestrian interaction behaviors and movement adaptations, we extract several quantitative metrics for each individual and pedestrian pair:

- **Minimum Encounter Distance:** Closest Euclidean distance between two agents during an interaction episode, which captures the spatial proximity that may trigger behavioral responses.
- **Speed Change (Δv):** Difference in average velocity between immediately before and after an interaction, which indicates acceleration or deceleration adjustments in response to surrounding pedestrians.
- **Heading Change ($\Delta\theta$):** Variation in motion direction computed from consecutive heading angles, which reflects navigational adjustments to avoid collisions or maintain group cohesion.
- **Trajectory Straightness:** Ratio of the direct Euclidean distance between the start and end points of a trajectory to its actual path length, which is a measure of movement efficiency and deviation caused by interactions.
- **Clearance Radius:** Instantaneous distance maintained from neighboring pedestrians calculated per timestamp, which captures local personal space preferences and crowding effects.

Interactions are further categorized based on social configuration:

- **S–S:** Single-to-single pedestrian encounters.
- **S–G:** Single pedestrian interactions with a group.

- **G–G:** Group-to-group interactions, representing collective dynamics.

All metrics are computed for each interaction event and aggregated by interaction type and discrete time bins. This aggregation enables both quantitative analysis of pedestrian behavior and qualitative examination of outliers or unusual responses.

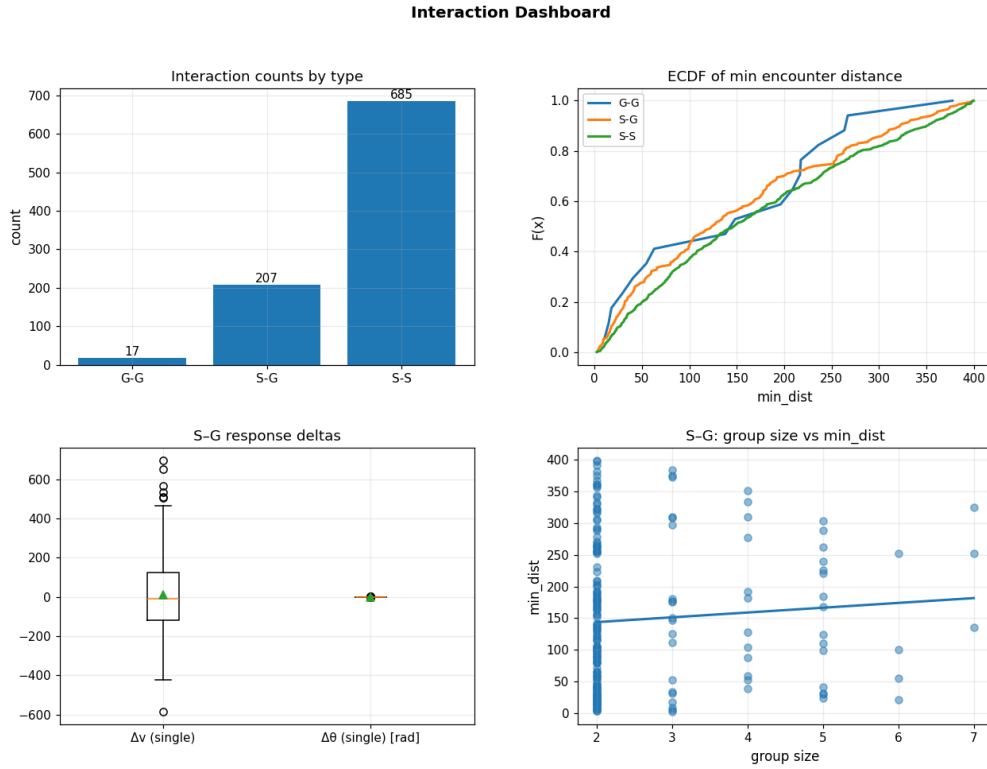


Figure 5.2: Interaction Dashboard (Number of data points = 60).

In Figure 5.2:

- **Top-left:** Histogram showing the frequency of interaction events by type — group–group (G–G), single–group (S–G), and single–single (S–S).
- **Top-right:** Empirical cumulative distribution function (ECDF) of minimum encounter distances for each interaction type, highlighting typical spatial proximities.
- **Bottom-left:** Boxplots of Δv (speed change) and $\Delta \theta$ (angular change in radians) for single pedestrians during S–G interactions to illustrate variability in movement adjustments.
- **Bottom-right:** Scatter plot with regression line showing the relationship between group size and minimum encounter distance in S–G interactions to show how group size influ-

ences pedestrian spacing.

Integrating both single and group pedestrian modeling provides a more realistic representation of social navigation dynamics, improves the interpretability of motion patterns, and enables analysis of pedestrian behavior at multiple social scales, from individual avoidance to coordinated group movement.

Chapter 6

Conclusion and Future Work

6.1 Achievements

This research introduced a comprehensive framework for calibrating and evaluating pedestrian agent-based models using data-driven optimization, multi-level feature extraction, and similarity-based trajectory comparison. The main goal was to establish a method capable of capturing heterogeneous, individual-level pedestrian behaviors and aligning simulated outcomes with empirical observations obtained from real-world environments. In the various modules—covering feature extraction, parameter reference generation, TabNet-based behavioral prediction, PSO calibration, and simulation execution—the proposed pipeline demonstrated that realistic, individualized pedestrian behavior can be reliably reconstructed, simulated, and validated.

A central achievement of this study was the successful implementation of a Particle Swarm Optimization (PSO)–driven calibration procedure that operates at the individual pedestrian level. By tuning key behavioral parameters, including speed bounds, acceleration–deceleration tendencies, turning-change percentages, and local sensitivity codes, the framework generates agent-specific parameter sets that closely reflect real pedestrian movement profiles. This calibration mechanism accommodates heterogeneity rather than relying on population-wide averages, leading to agents that behave in accordance with their unique empirical motion characteristics. The individualized calibration approach not only gives better micro-level behavioral realism

but also better fidelity of the emergent collective patterns at the environment scale.

Another major contribution relates to the introduction of similarity-based evaluation metrics for validating simulation results. By employing dynamic time warping (DTW), angle-difference metrics, and multidimensional comparison of velocity, position, and heading, the framework directly measures how closely simulated trajectories match their real counterparts. This data-driven validation establishes a principled method for quantifying the performance of behavioral models and identifying misalignments between simulated and empirical phenomena. The validation results confirmed that the calibrated model is capable of reproducing key temporal and spatial characteristics of individual pedestrian trajectories.

In addition to methodological advances, the framework demonstrated practical value for design validation and spatial analysis applications. By embedding calibrated agents into the reconstructed environment, the system enables simulation-based evaluation of how individuals interact with architectural configurations, bottlenecks, and spatial constraints. The integration of space-utilization heatmaps, pedestrian flow indicators, and interaction-based metrics provides new means of examining how specific design decisions influence movement efficiency and comfort. This positions the framework as a valuable tool for urban planning, evacuation analysis, public-space management, and pedestrian facility design.

In summary, the research has established (1) a scalable data-driven pipeline for pedestrian behavior reconstruction, (2) a robust optimization-based calibration methodology, and (3) a validation strategy capable of quantifying similarity between simulated and real pedestrian movement. Together, these contributions form a solid foundation for more-advanced simulation approaches that incorporate richer behavioral dynamics and multi-agent interactions.

6.2 Future Potential Extensions

While the present study focused primarily on individual-level pedestrian behavior, real-world pedestrian environments are fundamentally social, involving interactions between individuals, dyads, and groups. Extending the proposed framework to incorporate group-level modeling represents a significant and promising direction for future work. Groups exhibit behaviors

not reducible to individual actions, such as spatial cohesion, coordinated turning, social spacing preferences, and shared velocity adaptation. Incorporating these dynamics would require introducing group identifiers, extracting group-level behavioral features, and defining new parameters capturing intra-group attraction, turning consistency, and collective decision-making. As part of the existing PSO-based calibration system, these group-level parameters could be optimized to match empirical group behavior, providing a unified model capable of representing both single and group pedestrians simultaneously. This extension would substantially improve realism, particularly in environments dominated by families, tourists, or other socially connected groups.

Another promising direction is expanding the validation strategy to incorporate a broader set of empirical metrics and multi-agent performance indicators. The current similarity-based approach, while effective for trajectory-level comparisons, could be complemented with additional measures, such as collision-avoidance efficiency, local density response, space utilization distribution, and group formation stability. Integrating metrics for these would allow more nuanced evaluation of how well the calibrated model reproduces complex behavioral phenomena, especially under varying crowd densities or environmental configurations. Moreover, employing cross-environment validation (testing calibrated parameters on scenes different from the original training environment) could provide insights into the generalizability and transferability of the learned behavioral parameters.

Additionally, the framework could be extended to support real-time or near-real-time applications. By using the trained TabNet model and pre-calibrated agent parameters, the system could dynamically update pedestrian behaviors as new environmental or crowd information becomes available. Such adaptive simulations could be useful for intelligent crowd monitoring, predictive congestion management, and interactive urban design scenarios where stakeholders iteratively adjust environmental elements based on simulated outcomes.

Finally, the reconstruction of three-dimensional pedestrian motion from two-dimensional trajectories, introduced as a subsequent research direction, promises significant analytical and visualization advancements. Specifically, incorporating height, face orientation, and 3D spatial constraints could enrich behavioral interpretation and support more immersive simulation plat-

forms. This would enable more realistic rendering of pedestrian paths and deeper investigation into how visual attention, body posture, and inter-agent visibility shape movement decisions.

In summary, the present work laid a strong foundation for data-driven pedestrian simulation and opens multiple avenues for future development. Extending the framework to model group pedestrians, enhancing empirical validation strategies, exploring cross-environment generalization, and transitioning to multi-dimensional real-time simulation all represent meaningful next steps. Together, these efforts will create a comprehensive, flexible, and empirically grounded modeling ecosystem capable of capturing pedestrian behavior at multiple social and spatial scales.

Bibliography

- [1] The World Bank. Urban development: Overview. <https://www.worldbank.org/en/topic/urbandevelopment/overview>, 2025.
- [2] UN Trade and Development (UNCTAD). The unctad handbook of statistics 2023, 2023.
- [3] Dirk Helbing. Traffic and related self-driven many-particle systems. *Reviews of Modern Physics*, 73(4):1067–1141, 2001.
- [4] World Health Organization. Global status report on road safety 2023. <https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023>, 2023.
- [5] Mehdi Moussaïd, Dirk Helbing, and Guy Theraulaz. How simple rules determine pedestrian behavior and crowd disasters. *Proceedings of the National Academy of Sciences*, 108(17):6884–6888, 2011.
- [6] Todd Alexander Litman. *Evaluating Transportation Equity: Guidance for Incorporating Distributional Impacts in Transport Planning*. Victoria Transport Policy Institute, 2020.
- [7] Stephen Xia, Daniel de Godoy Peixoto, Bashima Islam, Md Tamzeed Islam, Shahriar Nirjon, and Peter R. Kinget. Improving pedestrian safety in cities using intelligent wearable systems. *IEEE Internet of Things*, 6(5):7497–7514, 2019.
- [8] C. Macal and M North. Tutorial on agent-based modelling and simulation. *Journal of Simulation*, 4(3):151–162, 2010.
- [9] Thomas C. Schelling. Dynamic models of segregation. *Journal of Mathematical Sociology*, 1(2):143–186, 1971.

- [10] John H. Holland and John H. Miller. Artificial adaptive agents in economic theory. *American Economic Review*, 81(2):365–371, 1991.
- [11] Volker Grimm, Uta Berger, Donald L. DeAngelis, J. Gary Polhill, Jarl Giske, Steven F. Railsback. The odd protocol: a review and first update. *Ecological Modelling*, 221(23):2760–2768, 2010.
- [12] Li An, Volker Grimm, Abigail Sullivan, B.L. Turner, Nicolas Malleson, Alison Heppens, Christian Vincenot, Derek Robinson, Xinyue Ye, Jianguo Liu, Emilie Lindkvist, and Wenwu Tang. Challenges, tasks, and opportunities in modeling agent-based complex systems. *Ecological Modelling*, 457:109685, 2021.
- [13] Rosaria Conte and Mario Paolucci. On agent-based modeling and computational social science. *Frontiers in Psychology*, Volume 5 - 2014, 2014.
- [14] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*, volume 21, pages 25–34, 1987.
- [15] Dirk Helbing and Péter Molnár. Social force model for pedestrian dynamics. *Physical Review E*, 51(5):4282–4286, 1995.
- [16] Victor J. Blue and Jeffrey L. Adler. Cellular automata microsimulation for modeling bi-directional pedestrian walkways. *Transportation Research Part B: Methodological*, 35(3):293–312, 2001.
- [17] Carsten Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295(3–4):507–525, 2001.
- [18] Dirk Helbing, Illés J Farkas, and Tamás Vicsek. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, 2000.

- [19] Mehdi Moussaïd, Dirk Helbing, and Guy Theraulaz. How simple rules determine pedestrian behavior and crowd disasters. *Proceedings of the National Academy of Sciences*, 108(17):6884–6888, 2011.
- [20] Peng Zhang, Xiao-Xia Jian, and Bing-Hong Wang. Potential field cellular automata model for pedestrian flow. *Physical Review E*, 85(2):021119, 2012.
- [21] Jun Li, Siyao Fu, Haibo He, Hongfei Jia, Yanzhong Li, and Yi Guo. Simulating large-scale pedestrian movement using ca and event-driven model: Methodology and case study. *Physica A: Statistical Mechanics and its Applications*, 437:304–321, 2015.
- [22] W. G. Weng, T. Chen, H. Y. Yuan, and W. C. Fan. Cellular automaton simulation of pedestrian counter flow with different walk velocities. *Phys. Rev. E*, 74:036102, Sep 2006.
- [23] Jur Van den Berg, Ming Lin, and Dinesh Manocha. The velocity-based approach to crowd simulation. *Computer Animation and Virtual Worlds*, 18(3-4):319–331, 2007.
- [24] Qiancheng Xu, Mohcine Chraïbi, Antoine Tordeux, and Jun Zhang. Generalized collision-free velocity model for pedestrian dynamics. *arXiv preprint arXiv:1908.10304*, 2019.
- [25] Xiangmin Hu, Tao Chen, Kaifeng Deng, and Guanning Wang. Effects of the direction and speed strategies on pedestrian dynamics. *Chaos*, 32(063140), 2022.
- [26] Jun Zhang, Haoran Cui, Mohcine Chraïbi, Hang Yu, and Weiguo Song. Velocity-based model for pedestrian dynamics considering direction preferences. *Physica A: Statistical Mechanics and its Applications*, 613(128444), 2023.
- [27] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(Suppl 3):7280–7287, 2002.
- [28] Kheir Al-Kodmany. Crowd management and urban design: New scientific approaches. *URBAN DESIGN International*, 18(4):282–295, 2013.
- [29] R. A. Saeed, Diego Reforgiato Recupero, and Paolo Remagnino. Modelling group dynamics for crowd simulations. *Personal and Ubiquitous Computing*, 26:1299–1319, 2022.

- [30] Shanwen Yang, Tianrui Li, Xun Gon, Bo Peng, and Jie Hu. A review on crowd simulation and modeling. *Graphical Models*, 111:101081, 2020.
- [31] Eleonora Papadimitriou, Sylvain Lassarre, and George Yannis. Human factors of pedestrian walking and crossing behaviour. *Transportation Research Procedia*, 25:2002–2015, 2017.
- [32] George Kouskoulis, Ioanna Spyropoulou, and Constantinos Antoniou. Pedestrian simulation: Theoretical models vs. data driven techniques. *International Journal of Transportation Science and Technology*, 7(4):241–253, 2018.
- [33] Weili Wang, Jiayu Rong, Qinqin Fan, Jingjing Zhang, Xin Han, and Beihua Cong. Data-driven simulation of pedestrian movement with artificial neural network. *Journal of Advanced Transportation*, 2021:5580910, 2021.
- [34] Xiao Song, Daolin Han, Jinghan Sun, and Zenghui Zhang. A data-driven neural network approach to simulate pedestrian movement. *Physica A: Statistical Mechanics and its Applications*, 509:827–844, 2018.
- [35] David J. Butts, Noelle E. Thompson, Sonja A. Christensen, David M. Williams, and Michael S. Murillo. Data-driven agent-based model building for animal movement through exploratory data analysis. *Ecological Modelling*, 470:110001, 2022.
- [36] Jinghui Zhong, Dongrui Li, Zhixing Huang, Chengyu Lu, and Wentong Cai. Data-driven crowd modeling techniques: A survey. *ACM Computing Surveys*, 32(1):1–33, 2022.
- [37] Bin Jiang and Tao Jia. Agent-based simulation of human movement shaped by the underlying street structure. *International Journal of Geographical Information Science*, 25(1):51–64, 2011.
- [38] Alexander Michelsa, Jeon-Young Kangb, and Shaowen Wangc. Particle swarm optimization for calibration in spatially explicit agent-based modeling. *Journal of Artificial Societies and Social Simulation*, 25(2):8, 2022.

- [39] Shuhan Du, Wenhui Fan, and Yi Liu. A novel multi-agent simulation based particle swarm optimization algorithm. *PLoS ONE*, 17(10):e0275849, 2022.
- [40] Yaochu Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.
- [41] Yuan Tian, Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink. Real-time model calibration with deep reinforcement learning. *Mechanical Systems and Signal Processing*, 165:108284, 2022.
- [42] Ruhollah Jamali and Sanja Lazarova-Molnar. A comprehensive framework for data-driven agent-based modeling. In *2024 Winter Simulation Conference (WSC)*, pages 620–631, 2024.
- [43] Haifeng Zhang, Yevgeniy Vorobeychik, Joshua Letchford, and Kiran Lakkaraju. Data-driven agent-based modeling, with application to rooftop solar adoption. *Autonomous Agents and Multi-Agent Systems*, 30(6):1023–1049, 2016.
- [44] Liang Chen. Agent-based modeling in urban and architectural research: A brief literature review. *Frontiers of Architectural Research*, 1(2):166–177, 2012.
- [45] Alison Heppenstall, Andrew Crooks, Nick Malleson, Ed Manley, Jiaqi Ge1, and Michael Batty. Future developments in geographical agent-based models: Challenges and opportunities. *Geographical Analysis*, 53:76–91, 2021.
- [46] David Stieler, Tobias Schwinn, Samuel Leder, Mathias Maierhofer, Fabian Kannenberg, and Achim Menges. Agent-based modeling and simulation in architecture. *Automation in Construction*, 141:104426, 2022.
- [47] Kostas Cheliotis. An agent-based model of public space use. *Computers, Environment and Urban Systems*, 81:101476, 2020.
- [48] Daniil Voloshin, Dmitriy Rybokonenko, and Vladislav Karbovskii. Optimization-based calibration for micro-level agent-based simulation of pedestrian behavior in public spaces. *Procedia Computer Science*, 66:372–381, 2015.

- [49] MD Jahedul Alam, Muhammad Ahsanul Habib, and Devin Holmes. Pedestrian movement simulation for an airport considering social distancing strategy. *Transportation Research Interdisciplinary Perspectives*, 13:100527, 2022.
- [50] Manon Prédhumeau, Lyuba Mancheva, Julie Dugdale, and Anne Spalanzani. An agent-based model to predict pedestrians trajectories with an autonomous vehicle in shared spaces. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 1010–1018. IFAAMAS, 2021.
- [51] Y. Tao, Z. Li, Y. Liu, H. Zhu, and L. Wu. A comparative analysis of trajectory similarity measures. *GIScience & Remote Sensing*, 58(5):643–669, 2021.
- [52] D. Brscic, T. Kanda, T. Ikeda, and T. Miyashita. Person position and body direction tracking in large public spaces using 3d range sensors. *IEEE Transactions on Human-Machine Systems*, 43(6):522–534, 2013.
- [53] Russell Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43. IEEE, 1995.
- [54] Yuhui Shi and Russell C. Eberhart. A modified particle swarm optimizer. In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360)*, pages 69–73. IEEE, 1998.
- [55] Sercan O. Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. *arXiv preprint arXiv:1908.07442*, 2019. v5, last revised 9 Dec 2020.
- [56] Amartaivan Sanjjamts and Hiroshi Morita. Preliminary study on space utilization and emergent behaviors of group vs. single pedestrians in real-world trajectories. *arXiv preprint arXiv:2508.18939*, 2025. Computer Vision and Pattern Recognition (cs.CV); Applications (stat.AP).
- [57] Siyu Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Understanding pedestrian behaviors from stationary crowd groups. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Appendix

Appendix A: Daily Summary of ATC Trajectory Data

The dataset used in this study is available at the [ATC Dataset page](#)

Table 1: Summary statistics of daily ATC pedestrian trajectory data. Each file corresponds to one day of observation (Part 1, 43 files).

File Name	Total Records	Total Agents	Min Records / Agent	Max Records / Agent	Avg Records / Agent	Rec. Freq. (sec)	Total Duration (hours)
atc-20121024.csv	16817748	17558	6	65662	957.8396173	0.044061999	11.11758333
atc-20121028.csv	40957068	49421	1	358293	828.7381478	0.062104423	11.2351325
atc-20121031.csv	18610827	18435	10	114383	1009.537673	0.039756382	11.77751444
atc-20121104.csv	37055936	44181	9	66471	838.7301329	0.048727008	11.18934833
atc-20121107.csv	21045939	17657	1	117914	1191.931755	0.048241214	11.01992333
atc-20121111.csv	41633006	45357	1	99112	917.8959367	0.060815525	10.76135694
atc-20121114.csv	15606040	18145	1	136449	860.0738495	0.044431313	10.88367889
atc-20121118.csv	42667343	51936	1	75128	821.5369493	0.049625052	10.91146306
atc-20121121.csv	25887021	22227	1	200951	1164.665542	0.040238086	11.58943306
atc-20121125.csv	47398149	56036	1	62180	845.851756	0.05352981	11.06586056
atc-20121128.csv	16814060	16617	7	127489	1011.85894	0.046350755	11.03903361
atc-20121202.csv	39909589	49163	5	49323	811.7809938	0.051944158	10.66510139
atc-20121205.csv	26885731	19760	2	198085	1360.613917	0.05181493	10.97191639
atc-20121209.csv	41269159	47814	1	109029	863.1187309	0.085687971	10.32750528
atc-20121212.csv	23765551	20759	1	197644	1144.831206	0.047720386	11.53919
atc-20121216.csv	60353896	70902	1	91787	851.2298102	0.062744251	11.30594806
atc-20121219.csv	16900032	17734	2	196917	952.9734972	0.039584364	10.67246917
atc-20121223.csv	45897966	47472	1	123639	966.8428969	0.065147376	11.10409
atc-20121226.csv	27611555	27102	1	180526	1018.80138	0.039905952	11.77517278
atc-20130106.csv	56000937	80988	1	224164	691.4720329	0.075292984	10.67920306
atc-20130109.csv	23394238	19760	1	197013	1183.918927	0.038222674	12.13714083
atc-20130113.csv	43365026	47399	1	129779	914.8932678	0.049746196	10.77873722
atc-20130116.csv	32646986	20122	1	378271	1622.452341	0.044710887	11.19123778
atc-20130120.csv	40793112	43487	1	95407	938.0530273	0.049432857	10.90785083
atc-20130123.csv	32795959	19235	1	518781	1705.014765	0.042347569	10.59049861
atc-20130127.csv	42107176	47882	1	64251	879.3946786	0.05147082	11.53162528
atc-20130130.csv	26222053	19973	1	168959	1312.875031	0.046341106	10.93581417
atc-20130203.csv	46638183	44596	1	247626	1045.792963	0.051043029	10.71713278
atc-20130206.csv	24735816	22576	1	154012	1095.668675	0.046012028	12.7798125
atc-20130210.csv	43594977	57733	3	60749	755.1136612	0.073630972	10.8757775
atc-20130213.csv	17120056	17533	9	123814	976.4476131	0.043331905	11.04436
atc-20130217.csv	47306214	50667	1	187766	933.6691338	0.0614915	10.86211917
atc-20130220.csv	26296575	22244	10	339410	1182.187331	0.04164062	13.16522111
atc-20130224.csv	32742929	40374	2	59942	810.9904642	0.048989912	10.96842694
atc-20130227.csv	24234025	19561	8	102845	1238.894995	0.043092746	10.69389722
atc-20130303.csv	39336656	46113	1	53652	853.0491618	0.059491081	9.775237222
atc-20130306.csv	28622570	25227	1	130248	1134.600626	0.047443435	11.33042861
atc-20130310.csv	51977222	74087	1	119933	701.570073	0.072409815	11.03306722
atc-20130317.csv	50746576	66117	1	84336	767.5268993	0.088253629	10.78217556
atc-20130320.csv	40548787	42541	1	150130	953.1695776	0.058962332	10.88659694
atc-20130324.csv	42192073	52168	1	122078	808.7730601	0.064954621	11.50424361
atc-20130327.csv	27842563	29784	1	301186	934.8161093	0.046871992	13.54394278
atc-20130331.csv	48322615	62375	1	132489	774.7112625	0.057577887	11.02202417

Table 2: Summary statistics of daily ATC pedestrian trajectory data. Each file corresponds to one day of observation (Part 2, 49 files, continued).

File	Total Records	Total Agents	Min Records / Agent	Max Records / Agent	Avg Records / Agent	Rec. Freq. (sec)	Total Duration (hours)
atc-20130403.csv	26962158	24981	4	186768	1079.306593	0.041148349	13.55059222
atc-20130407.csv	48646502	60546	1	59951	803.4635153	0.056898716	10.78192722
atc-20130410.csv	20554565	18783	1	131711	1094.317468	0.041255188	10.67152278
atc-20130414.csv	57799959	62082	1	60225	931.0260462	0.046567773	11.00706194
atc-20130417.csv	21806989	15893	1	159043	1372.112817	0.033116231	11.1193
atc-20130421.csv	57765681	60352	1	128017	957.1460929	0.040031379	11.34075194
atc-20130424.csv	16336530	13821	2	190753	1182.007814	0.034635618	10.6930175
atc-20130428.csv	63747537	87136	1	181723	731.5866806	0.069320029	11.11284528
atc-20130501.csv	52695220	44790	1	197357	1176.4952	0.034208705	10.486355
atc-20130505.csv	66702871	85290	1	173252	782.0714152	0.057725468	11.02834972
atc-20130508.csv	20969838	16373	4	125513	1280.757222	0.034980064	10.80257
atc-20130512.csv	54561354	46346	1	74448	1177.261339	0.035662861	10.59467583
atc-20130515.csv	31832330	18921	1	388122	1682.380952	0.033148468	10.46524417
atc-20130519.csv	53463234	46587	1	166722	1147.599845	0.041423068	11.20005278
atc-20130522.csv	40090427	23790	1	473126	1685.179781	0.033193992	10.46657667
atc-20130526.csv	49065213	40528	3	149480	1210.649748	0.035056295	11.38504194
atc-20130529.csv	37081781	20129	1	305520	1842.206816	0.033277508	10.52868389
atc-20130602.csv	63489264	59333	1	138435	1070.049787	0.046129564	11.1240375
atc-20130605.csv	30462981	20051	1	316550	1519.274899	0.033247799	11.39477083
atc-20130609.csv	45892920	41758	2	80609	1099.021026	0.038615285	11.12564972
atc-20130612.csv	29778354	19658	1	186173	1514.821142	0.03440927	11.19033306
atc-20130616.csv	44124560	39665	1	86391	1112.430606	0.051464339	9.757671944
atc-20130619.csv	22504880	17537	1	117016	1283.279922	0.033125371	10.6234275
atc-20130623.csv	48392824	41205	3	266755	1174.440578	0.03763971	11.28358806
atc-20130626.csv	25980692	15623	1	256983	1662.977149	0.033191289	10.76097111
atc-20130630.csv	52248696	48219	1	145746	1083.570709	0.050521953	11.30852
atc-20130703.csv	21335575	13604	1	342079	1568.331006	0.033218628	9.880096111
atc-20130707.csv	58223598	51324	1	79193	1134.432195	0.042209107	10.67714
atc-20130710.csv	42270276	26638	3	283174	1586.841204	0.034565774	10.56524306
atc-20130714.csv	59372301	56378	1	226786	1053.11116	0.048178284	11.23833583
atc-20130717.csv	24884247	16575	2	600072	1501.312036	0.03464681	10.54300806
atc-20130721.csv	66472776	66638	1	215927	997.5205738	0.054884948	10.39263583
atc-20130724.csv	42055121	30739	2	261679	1368.135626	0.035032432	10.32570917
atc-20130728.csv	40982223	57602	4	199977	711.4722232	0.081283269	10.27201694
atc-20130807.csv	47045115	39758	2	249666	1183.28676	0.037328685	10.72050917
atc-20130811.csv	38877481	57270	1	68247	678.8454863	0.110670729	11.06155917
atc-20130814.csv	64421063	56384	1	401298	1142.541554	0.043805906	10.5312
atc-20130818.csv	61373905	69652	1	172870	881.1506489	0.063469113	10.80148861
atc-20130821.csv	42637449	29143	1	172108	1463.042549	0.034470459	10.92288639
atc-20130825.csv	59646402	66643	1	128383	895.0137599	0.051049719	11.91409806
atc-20130828.csv	44456064	28935	3	214681	1536.411405	0.040054547	10.90069667
atc-20130901.csv	62556324	66311	1	129162	943.3777805	0.050708739	11.04396083
atc-20130904.csv	19904958	13368	6	350048	1489.000449	0.035025408	10.74714
atc-20130911.csv	28994849	19767	1	359384	1466.831032	0.033398086	10.71244278
atc-20130915.csv	54714277	52518	1	186869	1041.81951	0.043241674	10.93997472
atc-20130918.csv	28211549	17831	1	205660	1582.163031	0.033137419	10.95305556
atc-20130922.csv	57047008	47833	1	173533	1192.628687	0.037298184	10.78954111
atc-20130925.csv	31344759	20173	2	577738	1553.797601	0.034159141	10.53692528
atc-20130929.csv	58292827	53048	1	283399	1098.869458	0.041865672	10.54565056
AVERAGE	39,508,124	38,895	1	600,072	1,112.60	0.048	11.02

Appendix B: Algorithm for Standard PSO-based Calibration

Algorithm 4: Standard PSO-based Model Calibration

Input: Agent data $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$, Environmental conditions $\mathcal{E}$, PSO parameters (c_1, c_2, w) , Max iterations I_{max} , Number of particles P

Output: Optimized tuning parameters Θ^* for each agent

Initialize:

- Randomly initialize P particles with tuning parameters $\Theta = \{\theta_1, \theta_2, \dots, \theta_P\}$ for each agent $a \in \mathcal{A}$
- Set particle velocities $v_p = 0$ for all particles
- Set global best position $g_{best} \leftarrow \infty$
- Set individual best positions $p_{best} \leftarrow \infty$ for all particles

foreach agent $a \in \mathcal{A}$ **do**

Agent-Specific Optimization:

 Load environment condition $\mathcal{E}_a$ for agent a

for $t = 1$ to I_{max} **do**

foreach particle $p \in P$ **do**

Evaluate Objective Function:

 Run full simulation with parameters θ_p under $\mathcal{E}_a$ to obtain simulated velocity $\hat{v}_p$

 Compute error $f_p \leftarrow \text{Objective Function}(\hat{v}_p, \text{real velocity}_a)$

Update Personal and Global Best:

if $f_p < p_{best}[p]$ **then**

$p_{best}[p] \leftarrow f_p$

$p_{best_pos}[p] \leftarrow \theta_p$

end

if $f_p < g_{best}$ **then**

$g_{best} \leftarrow f_p$

$g_{best_pos} \leftarrow \theta_p$

end

end

Update Particle Velocities and Positions:

foreach particle $p \in P$ **do**

$v_p \leftarrow w \cdot v_p + c_1 \cdot r_1 \cdot (p_{best_pos}[p] - \theta_p) + c_2 \cdot r_2 \cdot (g_{best_pos} - \theta_p)$

$\theta_p \leftarrow \theta_p + v_p$

end

end

end

return optimized parameter set Θ^* for each agent obtained through simulation-based evaluation.

Appendix C: Detailed Experimental Information

Table 3: Detailed list of sensitivity analysis experiments (Part 1: Instances 1–48).

# Agents	ID	View Angle	Wall Code	Crowd Code	PSO Iter	Particles	Runtime	Calibration Quality (Best, Worst, Mean)
100	1	[-30,30]	{0,1,2,3}	{0,1,2,3}	10	5	6.16	15.99, 14.62, 15.21
100	2	[-30,30]	{0,1,2,3}	{0,1,2,3}	10	10	7.65	18.62, 15.01, 17.18
100	3	[-30,30]	{0,1,2,3}	{0,1,2,3}	50	5	22.17	24.23, 21.02, 22.24
100	4	[-30,30]	{0,1,2,3}	{0,1,2,3}	50	10	26.96	27.95, 21.24, 24.19
100	5	[-30,30]	{0,1,2,3}	{0,1,2,3}	100	5	38.76	54.89, 42.18, 49.32
100	6	[-30,30]	{0,1,2,3}	{0,1,2,3}	100	10	43.12	56.23, 44.02, 50.21
100	7	[-30,30]	{0,1,2,3}	{0,1,2}	10	5	4.02	17.69, 13.26, 15.10
100	8	[-30,30]	{0,1,2,3}	{0,1,2}	10	10	5.68	19.02, 14.97, 16.09
100	9	[-30,30]	{0,1,2,3}	{0,1,2}	50	5	17.69	24.15, 20.82, 22.64
100	10	[-30,30]	{0,1,2,3}	{0,1,2}	50	10	24.01	30.89, 23.02, 26.02
100	11	[-30,30]	{0,1,2,3}	{0,1,2}	100	5	36.02	53.85, 47.05, 48.09
100	12	[-30,30]	{0,1,2,3}	{0,1,2}	100	10	39.01	55.94, 45.98, 51.47
100	13	[-30,30]	{0,1,2}	{0,1,2,3}	10	5	5.06	21.52, 18.05, 19.75
100	14	[-30,30]	{0,1,2}	{0,1,2,3}	10	10	6.78	20.61, 16.18, 17.58
100	15	[-30,30]	{0,1,2}	{0,1,2,3}	50	5	18.12	26.05, 21.92, 23.14
100	16	[-30,30]	{0,1,2}	{0,1,2,3}	50	10	22.14	31.98, 22.36, 25.71
100	17	[-30,30]	{0,1,2}	{0,1,2,3}	100	5	35.67	54.21, 46.98, 49.54
100	18	[-30,30]	{0,1,2}	{0,1,2,3}	100	10	40.39	56.12, 47.12, 50.91
100	19	[-30,30]	{0,1,2}	{0,1,2}	10	5	3.28	20.12, 16.05, 17.84
100	20	[-30,30]	{0,1,2}	{0,1,2}	10	10	5.31	22.38, 18.11, 19.98
100	21	[-30,30]	{0,1,2}	{0,1,2}	50	5	16.62	27.69, 22.12, 24.97
100	22	[-30,30]	{0,1,2}	{0,1,2}	50	10	20.19	30.71, 26.36, 27.65
100	23	[-30,30]	{0,1,2}	{0,1,2}	100	5	32.14	56.65, 48.98, 53.11
100	24	[-30,30]	{0,1,2}	{0,1,2}	100	10	38.26	59.12, 50.12, 54.23
100	25	[-60,60]	{0,1,2,3}	{0,1,2,3}	10	5	5.64	14.01, 12.03, 12.95
100	26	[-60,60]	{0,1,2,3}	{0,1,2,3}	10	10	8.11	17.26, 13.98, 14.68
100	27	[-60,60]	{0,1,2,3}	{0,1,2,3}	50	5	21.96	24.65, 20.18, 21.06
100	28	[-60,60]	{0,1,2,3}	{0,1,2,3}	50	10	26.27	26.46, 20.65, 23.43
100	29	[-60,60]	{0,1,2,3}	{0,1,2,3}	100	5	39.04	53.12, 41.96, 48.01
100	30	[-60,60]	{0,1,2,3}	{0,1,2,3}	100	10	44.01	55.95, 43.11, 48.13
100	31	[-60,60]	{0,1,2,3}	{0,1,2}	10	5	4.69	17.50, 14.03, 15.16
100	32	[-60,60]	{0,1,2,3}	{0,1,2}	10	10	5.14	18.61, 14.12, 15.64
100	33	[-60,60]	{0,1,2,3}	{0,1,2}	50	5	18.02	23.96, 21.05, 22.16
100	34	[-60,60]	{0,1,2,3}	{0,1,2}	50	10	24.74	29.74, 23.61, 26.61
100	35	[-60,60]	{0,1,2,3}	{0,1,2}	100	5	36.02	52.19, 45.12, 47.08
100	36	[-60,60]	{0,1,2,3}	{0,1,2}	100	10	40.11	53.15, 44.02, 50.11
100	37	[-60,60]	{0,1,2}	{0,1,2,3}	10	5	4.81	20.12, 15.21, 16.79
100	38	[-60,60]	{0,1,2}	{0,1,2,3}	10	10	6.11	19.82, 16.21, 17.19
100	39	[-60,60]	{0,1,2}	{0,1,2,3}	50	5	18.63	23.05, 20.85, 22.96
100	40	[-60,60]	{0,1,2}	{0,1,2,3}	50	10	21.86	30.19, 21.19, 24.03
100	41	[-60,60]	{0,1,2}	{0,1,2,3}	100	5	36.79	53.36, 45.01, 47.91
100	42	[-60,60]	{0,1,2}	{0,1,2,3}	100	10	41.57	54.65, 45.98, 51.04
100	43	[-60,60]	{0,1,2}	{0,1,2}	10	5	3.11	19.62, 14.13, 15.95
100	44	[-60,60]	{0,1,2}	{0,1,2}	10	10	4.91	20.61, 18.36, 19.14
100	45	[-60,60]	{0,1,2}	{0,1,2}	50	5	17.15	28.01, 21.91, 24.12
100	46	[-60,60]	{0,1,2}	{0,1,2}	50	10	21.01	31.06, 25.31, 26.96
100	47	[-60,60]	{0,1,2}	{0,1,2}	100	5	33.05	54.15, 46.18, 50.98
100	48	[-60,60]	{0,1,2}	{0,1,2}	100	10	39.18	57.32, 49.88, 52.05

The experiments listed in Table 3 consist of multiple instances, each corresponding to a unique combination of environmental condition codes, design parameters, and PSO calibration settings. The crowd radius was set proportional to the agent’s *height*, and the wall and pillar threshold was set proportional to the agent’s *velocity*. The pillar code was treated as a single option set $0,1,2$. The maximum number of simulation steps was fixed at 1000, and the number of local condition records was fixed at 200.

Table 4: Detailed list of sensitivity analysis experiments (Part 2: Instances 49–57)

# Agents	ID	View Angle	Wall Code	Crowd Code	PSO Iter	Particles	Runtime	Calibration Quality (Best, Worst, Mean)
200	49	[-30,30]	{0,1,2}	{0,1,2}	100	10	77.03	57.91, 48.36, 52.61
500	50	[-30,30]	{0,1,2}	{0,1,2}	100	10	189.45	60.31, 49.15, 53.48
800	51	[-30,30]	{0,1,2}	{0,1,2}	100	10	308.13	58.68, 48.98, 53.08
1000	52	[-30,30]	{0,1,2}	{0,1,2}	100	10	379.36	60.69, 46.34, 53.12
100	53	[-30,30]	{0,1,2}	{0,1,2}	100	10	206.36	25.12, 18.31, 21.71
200	54	[-30,30]	{0,1,2}	{0,1,2}	100	10	294.84	26.78, 17.03, 19.94
500	55	[-30,30]	{0,1,2}	{0,1,2}	100	10	579.12	26.05, 14.61, 18.12
800	56	[-30,30]	{0,1,2}	{0,1,2}	100	10	866.24	25.17, 13.64, 20.56
1000	57	[-30,30]	{0,1,2}	{0,1,2}	100	10	1075.19	27.03, 13.31, 19.07

The experimental results listed in Table 4 were based on the configuration of *Instance 24* in Table 3.