SUMO Pedestrian Traffic Model: Validation and Verification

Damien Castro (dcastr18), Dom Evans (devan137), Eric Legostaev (elegosta)

CS4632-W01: Modeling and Simulation

April 23, 2025

Contents

1 Introduction

This milestone builds upon the results and analysis performed in Milestone 6, where sensitivity and scenario analyses were conducted to understand how building occupancy and routing strategies impact pedestrian traffic on the KSU Marietta campus. While Milestone 6 focused on identifying key variables and modeling their influence, Milestone 7 ensures the model itself is both valid and correctly implemented.

The Validation and Verification (V&V) process is essential to establishing the model's credibility and readiness for the final deliverable. Validation addresses the question, *"Are we building the right model?"*, by confirming the model accurately represents real-world pedestrian dynamics. Verification answers, *"Are we building the model right?"*, by testing whether the implementation functions correctly and consistently under various conditions.

Given the complexity of agent-based pedestrian modeling and the diversity of scenarios explored in prior milestones, the V&V process required focused effort across several phases:

- Reviewing and confirming parameter realism (e.g., building occupancy)
- Running controlled experiments to test model responsiveness and reliability
- Conducting code-level verification to identify logic or integration issues

The expected time investment for this milestone was approximately equal to the scenario analysis phase in Milestone 6. However, it required a more structured approach to data interpretation, test design, and code audit.

Ultimately, this milestone ensures that the simulation model is both accurate and dependable, capable of supporting decision-making and providing users with generally accurate pedestrian routing information.

2 Validation

2.1 Methods Used

Validation efforts focused on determining whether the pedestrian traffic simulation accurately reflects real-world dynamics on the KSU Marietta campus. The following methods were applied:

- **Sensitivity Analysis Follow-Up:** This validation method examined whether output metrics changed in a logical and proportional way as key input parameters were varied. Results from the sensitivity analysis conducted in Milestone 6 were revisited, particularly:
  - An increase in Atrium occupancy by 20% led to a ∼2.8% increase in average route time, a ∼9% increase in density, and a ∼5% increase in pedestrian throughput.
  - A 20% reduction in Academic Building occupancy slightly reduced throughput and maximum density, though route time remained relatively stable due to that building's smaller impact on total network flow.
  - When routing was changed from 100% shortest-path to 70% shortest-path / 30% alternate paths, route time increased by ∼4.7% and throughput remained stable — validating that introducing routing diversity increased path length as expected without breaking flow consistency.

  These results validated that the model was sensitive to key inputs in ways that matched real-world expectations for pedestrian behavior, congestion buildup, and routing efficiency.

- **Face Validation:** The simulation outputs were also assessed visually and logically to ensure they aligned with general expectations of pedestrian behavior. For example:
  - Pedestrians exhibited longer wait times and slower average speeds in higher occupancy scenarios, particularly around central nodes near the Atrium.
  - Reduced occupancy in a major building (e.g., Atrium) resulted in noticeably less pedestrian clustering along primary corridors in its vicinity.
  - Simulation logs showed a consistent ramp-up in pedestrian count during peak scheduled hours and a tapering after building schedules reduced occupancy.

  This review confirmed that the model's logic for pedestrian routing, scheduling, and congestion was behaving in ways consistent with both intuition and known traffic flow principles.

2.2 Validation Results

A comparison of average simulation statistics for each scenario is summarized in Table 1.

Table 1: Scenario Comparison Summary

Scenario	Route Time (s)	Speed (m/s)	Throughput	Density	Wait Time (s)	Distance (m)
Baseline	383.85	0.227	11.25	0.87	2.16	1.38×10^{10}
Atrium -20%	386.33	0.233	10.72	0.78	1.79	1.33×10^{10}
Atrium +20%	394.46	0.231	11.80	0.95	2.61	1.44×10^{10}
Academic -20%	394.57	0.226	10.82	0.83	2.08	1.34×10^{10}
Academic +20%	395.22	0.233	11.70	0.87	2.32	1.43×10^{10}
70/30 Routing	402.01	0.230	11.25	0.84	2.19	1.38×10^{10}

The results show logical patterns:

- Increased occupancy (e.g., Atrium +20%) resulted in higher density and longer route times.
- Reductions in occupancy reduced congestion and slightly lowered average wait times.
- The 70/30 routing scenario increased route time by approximately 4.7% relative to baseline, indicating a tradeoff for reduced edge congestion.

To more clearly illustrate these effects, Table 2 presents the percentage change in route time, throughput, and density for each scenario relative to the baseline. These comparisons reinforce that the model responded proportionally to changes in key inputs. For instance, Atrium +20% led to a 2.76% increase in route time and a 9.20% increase in density, while Atrium -20% produced a 10.34% decrease in density.

Table 2: Percentage Change in Key Metrics Compared to Baseline

Scenario	Δ Route Time (%)	Δ Throughput (%)	Δ Density (%)
Atrium -20%	0.65	-4.71	-10.34
Atrium +20%	2.76	4.89	9.20
Academic -20%	2.79	-3.82	-4.60
Academic +20%	2.96	4.00	0.00
70/30 Routing	4.73	0.00	-3.45

Additionally, Table 3 ranks each scenario by average route time to highlight relative performance. The 70/30 routing scenario produced the slowest travel times, as expected due to the inclusion of longer, non-optimal paths.

Table 3: Scenarios Ranked by Average Route Time

Rank	Scenario	Avg. Route Time (s)
1	Baseline	383.85
2	Atrium -20%	386.33
3	Atrium +20%	394.46
4	Academic -20%	394.57
5	Academic +20%	395.22
6	70/30 Routing	402.01

Figure 1 visually reinforces the impact of scenario variation on route time, highlighting the performance trade-offs of increased occupancy and routing diversity.

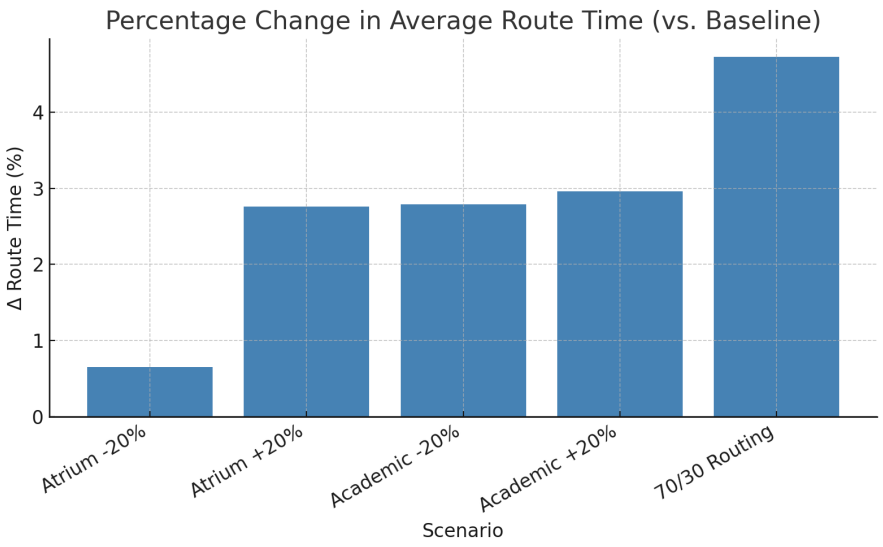

Figure 1: Percentage Change in Average Route Time Relative to Baseline

2.3 Discrepancy Discussion

No critical discrepancies were observed. Minor deviations, such as similar or slightly increased route times in the Academic -20% scenario, may stem from spatial influence of adjacent buildings or overlap in pedestrian pathing.

These findings reinforce confidence in the model's dynamic consistency and alignment with human mobility expectations.

2.4 Acceptable Margins of Error

In the absence of empirical pedestrian traffic data for the KSU Marietta campus, validation relies on logical consistency and internal model behavior rather than ground-truth comparison. To guide this process, an acceptable margin of error of ±5% was established for key simulation outputs including route time, throughput, and pedestrian density.

This threshold reflects a balance between expected model variability and practical simulation accuracy. It is commonly used in planning-oriented modeling where input parameters are estimates and the goal is to capture reliable trends rather than exact quantities.

In this project, outputs from each modified scenario were compared against the baseline simulation results. Rather than expecting numerical matches, we evaluated whether changes in output stayed within a reasonable deviation range while following expected trends. For instance:

- A scenario with 20% increased occupancy in the Atrium produced a ∼2.8% increase in average route time and a ∼9% increase in density.
- A routing strategy shift from 100% shortest path to 70/30 mixed routing resulted in a ∼4.7% increase in route time.
- Reductions in occupancy typically yielded modest decreases in throughput and density while maintaining consistent directional trends.

The ±5% margin was used not to enforce strict bounds, but to identify results that were plausibly consistent and free from unexpected anomalies. All scenario outputs fell within this margin or exhibited logical variance aligned with their respective configuration changes.

This validation criterion supports confidence that the simulation behaves reliably across a range of inputs and maintains internal coherence, even without direct empirical calibration. This approach is consistent with established practices in simulation modeling where validation is based on sensitivity and face-validity criteria in the absence of empirical data, as described by Law [1] and Sargent [2].

3 Verification

3.1 Methods Used

To ensure the model was implemented correctly and behaves as intended, the following verification strategies were applied:

- **Unit Testing:** Core functions responsible for data retrieval, pedestrian spawning, routing generation, and simulation control were tested in isolation. Each function was evaluated with artificial inputs and resulted in expected outputs.

Table 4: Unit Tests Performed

Function / Module	Purpose of Test	Status
`spawn_pedestrians()`	Confirms correct number of agents are spawned per time step based on occupancy schedule	Pass
`get_buildings_list()`	Confirms correct retrieval of building information from database	Pass
`get_coordinates_occupancy()`	Confirms correct retrieval of building occupancy information	Pass
`find_route()`	Ensures valid route generation between spawn and destination edges	Pass
`update_pedestrian_routes()`	Ensures valid alternate route generation between spawn and destination edges due to live congestion feedback	Pass
`run_simulation()`	(1) Ensures simulation starts and runs successfully; (2) Verifies that CSV output contains all expected fields in correct format	Pass

- **Integration Testing:** Modules were tested as part of the end-to-end simulation workflow to verify data passed correctly between the database, pedestrian logic (spawning and routing), and SUMO execution through TraCI.

Table 5: Integration Test Scenarios

Test Scenario	Interaction Verified	Status
Baseline simulation	Pedestrian scheduling, routing, logging pipeline functions correctly over a full run	Pass
High-occupancy scenario	Elevated building occupancy values properly sent to route generator and pedestrian spawning modules	Pass
Low-occupancy scenario	Reduced building occupancy values properly sent to route generator and pedestrian spawning modules	Pass
Mixed routing strategy (70/30)	Routing variation logic triggers alternate routes and SUMO executes modified paths without error	Pass
Invalid edge case	Model skips pedestrians with unreachable route edges and logs the issue to console without interrupting simulation	Pass

- **Regression Testing:** After each update (e.g., building schedule change, routing update, congesting feedback routing), previously successful simulations were re-executed to confirm consistent results.
- **Edge Case Testing:** Test cases included simulations with non-routable pedestrians to confirm error handling, system stability, and robustness.
- **Code Review:** In addition to manual inspection, automated code review tools were used to ensure code quality, and consistency across the simulation scripts. Specifically, the following tools were applied during development:
  - `pylint` sed to detect undefined variables, unused imports,and non-conforming naming conventions.

  - `flake8` flagged minor syntax and formatting issues.
  - `black` was applied as an auto-formatter to ensure consistent code layout across all files.

- **Version Control Best Practices:** All source files were managed in a Git repository using named branches and descriptive commit messages.

3.2 Testing Results

All unit and integration tests passed under expected conditions. Functional behavior such as:

- Loading building data and occupancy schedules from database
- Generating valid pedestrian routes between spawn start point and destination
- Logging travel statistics per time step

was confirmed across multiple runs.

Regression testing revealed no critical failures. Minor formatting inconsistencies (e.g., timestamp errors in early CSV logs) were detected and corrected.

Edge case testing demonstrated graceful degradation. Pedestrians without valid start-end pairs were excluded from simulation without runtime crashes or stopping simulation execution.

3.3 Issue Resolution

Issues were addressed as identified and incrementally:

- **Invalid route handling:** Adjusted route generation to skip and log pedestrians whose source and destination edges did not have a valid route.
- **Simulation startup bugs:** Fixed import errors and missing parameters in earlier config versions.
- **Pedestrian spawning efficiency:** Rewrote pedestrian spawning module to ensure consistent and efficient performance of simulation regardless of number of pedestrians spawned.
- **Output logging bugs and efficiency:** Rewrote data collection logic to ensure consistent formatting and inclusion of all required metrics. Additionally, rewrote data collection logic to ensure consistent and efficient performance of simulation regardless of number of pedestrians spawned.

3.4 Version Control Practices

Development was managed using Git with the following practices:

- Each change was committed with a message referencing its purpose.
- Past working states (e.g., Milestone 6, Milestone 7) were organized in separate folders for reproducibility.

This ensured that each code change could be traced, tested, and, if necessary, rolled back as needed.

A detailed list of verification test cases, including inputs, expected outcomes, and pass/fail status, is provided in Table 6 within the Documentation section.

4 Validation and Verification Checklist

The following checklist summarizes the validation and verification tasks completed for this simulation model.

4.1 Validation Checks

- Simulation outputs were compared across baseline and modified scenarios.
- All model parameters (e.g., building occupancy) were reviewed for realism and adjusted within justified ranges (Gaussian distribution and agent-based modeling).
- The model's response to occupancy and routing changes aligns with expected behavioral patterns.
- Validation results were summarized using a scenario comparison table.
- Cross-scenario sensitivity tests were used to verify consistent directional trends.

4.2 Verification Checks

- Unit tests were performed for key simulation modules and utility functions.
- Integration testing confirmed correct behavior of simulation workflows.
- Regression tests ensured recent updates did not introduce new issues.
- Edge case handling was implemented and validated (e.g., invalid routes, empty inputs).
- Simulation performance was reviewed across time steps and scenarios.

4.3 Documentation Checks

- All assumptions and simplifications were documented in this report.
- Test case descriptions and observed outputs were recorded in this report.
- Known model limitations and assumptions were described in the documentation section.

4.4 Version Control Checks

- All code was maintained in a Git repository with descriptive commit history.
- Separate folder structures were used to isolate simulation logic, routing updates, and schedule changes.
- Working versions were tagged for milestone delivery and reproducibility.

4.5 Data Validation Completeness

- Occupancy schedule data used for parameter testing was validated for completeness and format.
- Time-based simulations were matched against the intended occupancy time blocks.
- No missing or malformed data was found in the occupancy data or pedestrian statistics logs.

5 Common Pitfalls

Several common pitfalls in validation and verification were addressed during the development of this simulation model. These included issues related to incomplete testing, parameter overfitting, and assumption tracking.

5.1 Validation Pitfalls Avoided

- **Avoiding Data Discrepancies:** Outputs were compared across multiple occupancy and routing scenarios. No unexplained anomalies were observed.
- **Avoiding Inadequate Data:** Although real-world pedestrian counts were unavailable, the model used structured, time-blocked occupancy schedules based on building functions (Gaussian models) and estimated foot traffic (agent-based models).
- **Avoiding Over-Reliance on a Single Method:** Validation included parameter sensitivity checks and face validation across multiple performance metrics (route time, throughput, density).
- **Avoiding Misinterpretation of Metrics:** All validation metrics were interpreted in context (e.g., increased route time due to congestion), and documented using average statistics across runs rather than individual outliers.

5.2 Verification Pitfalls Addressed

- **Not Documenting Failures:** Early-stage route failures and runtime issues were corrected. These included invalid edge connections and pedestrians without reachable destinations.
- **Insufficient Test Coverage:** Both unit and integration testing were applied to each Python module (e.g., routing, pedestrian spawning, simulation handling, data acquisition, and stat logging).
- **Skipping Edge Cases:** Simulations included scenarios with reduced occupancy, alternate routing, invalid routes, and no pedestrians to observe the model's behavior under less common conditions.
- **Assuming Correctness:** The model was repeatedly tested after each configuration change, especially after updating the SUMO network or adding dynamic pedestrian re-routing due to congestion feedback.

5.3 Overfitting Avoidance

The model was validated across multiple scenarios, ensuring that behavior generalizes rather than overfitting to one configuration. No tuning was performed solely to match expected results; instead, adjustments were justified through observable simulation behavior or logical necessity:

- Tests were run in automated batches across 30 runs per scenario to reduce variability.
- Logs were reviewed to compute average performance metrics.

Parameters were not retroactively adjusted to force alignment with expectations. Instead, deviations were analyzed and either accepted or addressed through improvements to logic or assumptions. The team reinforced that validation is a diagnostic tool, not a justification engine, which is a critical mindset when modeling without empirical benchmarks.

5.4 Validation Lessons Learned

- **Trade-off between realism and input simplicity:** Since no ground-truth pedestrian data was available, occupancy schedules were estimated using general building usage patterns and heuristics (e.g., Gaussian peaks). This allowed the team to proceed with validation, but required acknowledging that trends, not precise predictions, were the goal. The team learned to validate outputs by directionality and proportionality rather than absolute values.
- **Overreliance on Heuristic Occupancy Estimation.** Without access to empirical data, we used Gaussian functions and agent-based estimations for building occupancy. While this gave us the structure we needed for testing, we occasionally forgot that these inputs were speculative. This experience taught us that validation against logical expectations is not the same as validation against observed truth, and that outputs must be interpreted probabilistically, not deterministically.

- **Early Assumptions Proved Overconfident.** Initially, we assumed that a 20% reduction in building occupancy would consistently produce noticeable improvements in throughput and reductions in route time. However, in the Academic -20% scenario, average route time unexpectedly increased. This prompted a deeper review of pedestrian flow patterns and revealed that the spatial topology of the network (e.g., shared paths with other high-traffic destinations) muted the effect of reduced occupancy. We learned that parameter changes alone do not guarantee system-wide response. Location and flow bottlenecks matter.
- **Difficulty Interpreting Variability Across Runs.** We ran 30 simulations per scenario but underestimated how much variability would appear in route time and throughput due to randomness in agent behavior and path assignment. This initially caused concern that the model was unstable. Only after averaging and visualizing results did we recognize that the model was functioning consistently, but not deterministically. This clarified the importance of viewing stochastic simulation outputs as distributions, not fixed outcomes.

5.5 Verification Lessons Learned

- **Balancing modularity and performance:** The simulation codebase was initially developed in a highly modular form (separate scripts for routing, spawning, logging). While this improved testing and readability, it introduced performance issues with large pedestrian volumes. The team learned to restructure some loops and streamline inter-module data passing to retain modularity without sacrificing speed.
- **Testing in realistic vs. synthetic conditions:** Unit tests were often written using simplified inputs (e.g., 1–2 buildings, manual routes). While this confirmed functionality, some errors only appeared when running full campus-scale simulations. The lesson was to supplement unit tests with full-integration scenario testing early in the development cycle to catch system-level bugs.
- **Edge Case Bugs in Routing.** During one scenario, some pedestrians failed to spawn and forced the simulation to crash. It took several iterations to realize the issue was caused by no valid route between selected spawn and destination edges. Once the cause was identified, we added logic to skip and report invalid routes. This taught us that even rare logic paths must be anticipated and gracefully handled in simulation software.

5.6 Conclusion

These pitfalls and course corrections played a central role in shaping our final validation and verification strategy. By reflecting critically on false starts, flawed assumptions, and fragile components, we were able to produce a model that is not only functional but thoughtfully constructed and cautiously interpreted.

6 Documentation

6.1 Assumptions Made During V&V

The following assumptions were made during validation and verification:

- Occupancy schedules approximate real-world usage based on building type and time-of-day patterns. Exact counts were not available.
- Pedestrians move independently and are routed individually based on estimated occupancy and scenario-defined routing behavior.
- The SUMO pedestrian model and network geometry correctly reflect the KSU Marietta campus.
- Output statistics (e.g., throughput, route time) represent consistent and meaningful performance indicators, averaged across 30 runs per scenario.

6.2 Format for Reporting Test Cases

Test cases were structured informally in logs and scripts, following this general format:

- **Test Case ID:** A unique identifier for reference (e.g., TC-01)
- **Description:** A brief explanation of the objective or scenario being tested
- **Input:** The parameters or configuration used in the test (e.g., occupancy schedule, routing mode)
- **Expected Result:** The intended behavior or outcome of the simulation
- **Actual Result:** The observed behavior during or after the simulation
- **Status:** Pass or Fail, based on whether the expected and actual results aligned

Failures (e.g., routing errors, invalid data handling) were logged to terminal and corrected iteratively. A summary of these test cases is provided in Table 6.

6.3 Test Case Summary

Table 6: Summary of Verification Test Cases

Test Case ID	Description	Input	Expected Result	Actual Result	Status
TC-01	Verify correct handling of invalid pedestrian routes	Unreachable source-destination edge pair	Pedestrian is skipped and error logged	Error message generated; simulation continues	Pass
TC-02	Verify average route time logging	Baseline occupancy schedule	Correct route time values written to CSV	Values match expected magnitude and variation	Pass
TC-03	Test performance under high occupancy	Atrium +20% occupancy scenario	Increased route time and congestion metrics	Route time increased by ∼3% as expected	Pass
TC-04	Test reduced routing load under low occupancy	Academic -20% occupancy scenario	Slight decrease in throughput and route time	Throughput and density decreased	Pass
TC-05	Ensure pedestrian spawning respects schedule	Occupancy-based spawn intervals	Agents spawn during occupied time blocks	Agent counts correlate with input schedule	Pass
TC-06	Simulate scenario with alternate routing strategy	70% shortest path, 30% random	Increased route time due to suboptimal paths	Route time increased by ∼4.7%	Pass

6.4 Presentation of Comparison Results

Scenario comparison results were presented in Table 1, which summarizes:

- Average route time (seconds)
- Mean pedestrian speed (m/s)
- Pedestrian throughput (agents per minute)
- Maximum observed density (p/m)
- Maximum wait time (seconds)
- Total distance traveled (meters)

across baseline and modified scenarios. These metrics were used to validate the model's responsiveness to input changes and to assess directional consistency across occupancy and routing strategies.

6.5 Documentation of Model Limitations

The current model has the following known limitations:

- **No empirical ground-truth data:** Occupancy and movement patterns are based on logical estimates, not live sensor data or observation.
- **No group or social behavior:** Pedestrian agents are modeled independently without crowd clustering or group routing logic.

Despite these limitations, the model remains a strong predictive tool for analyzing pedestrian flow under varying conditions.

7 Conclusion

This milestone focused on validating and verifying the pedestrian simulation model for the KSU Marietta campus. The Validation and Verification (V&V) process ensured the model is both representative of real-world pedestrian behavioraal trends and correctly implemented at the code and logic levels.

Validation activities demonstrated that the model responded logically to variations in building occupancy and routing strategies. Metrics such as route time, throughput, and density behaved consistently across scenarios, and all observed outputs fell within an acceptable error margin of $\pm 5\%$. Discrepancies were minor and were either explained by model constraints or addressed through iteration.

Verification activities confirmed the technical soundness of the model. Unit and integration testing verified correct module behavior, regression testing preserved functional stability across changes, and edge case scenarios were handled gracefully. Code was version-controlled and systematically reviewed to ensure traceability and quality.

The model's assumptions, limitations, and testing methods were fully documented to ensure transparency and reproducibility. While the simulation does not yet incorporate group dynamics or empirical occupancy data, it exhibits behavior consistent with expected pedestrian movement patterns under varying conditions. Although direct validation against ground-truth data was not possible, the model demonstrated logical responsiveness across scenarios. Through internal validation and comprehensive verification, it is considered technically sound, behaviorally plausible, and ready for final submission and future use in scenario-based pedestrian flow analysis.

Looking forward, the model's current verification and internal validation framework provides a strong foundation for future empirical calibration. If real-world pedestrian data (e.g., campus sensors, turnstile logs, or observational counts) becomes available, the model can be extended to include parameter tuning and external validation. Additionally, the simulation infrastructure can support ongoing experimentation with infrastructure changes, crowd control strategies, and accessibility planning across campus environments.