Reinforcement Learning Driven Warehouse Item Relocation Optimization

Abdelraheem Zekry, Arthur Noroes Reis, Binil Sajeev, Orlando Manrique
Matriculation Numbers: 22400668, 22409704, 22408791, 22403913
Department of Applied AI for Digital Production Management
Technische Hochschule Deggendorf, Campus Cham, Germany
Supervised by Prof. Dr. Hamidreza Heidari

***Abstract*—Warehouse slotting and item relocation are critical operational problems due to their direct impact on space utilization, picking efficiency, and labor cost. As warehouses grow in scale and complexity, traditional rule-based approaches struggle to balance competing objectives such as densification, accessibility, and ergonomic constraints. This study investigates the application of Reinforcement Learning (RL) to the warehouse item relocation problem under a fixed layout, where structural changes are not possible. A simulation-based framework is developed in which an RL agent is benchmarked against a deterministic heuristic baseline under identical physical, geometric, and operational constraints. Central to the approach is a shared physics-based allocation engine that guarantees geometric feasibility and safety compliance for all decisions. Using synthetic data, the RL agent learns relocation strategies that improve volumetric utilization while maintaining or reducing picking-related travel distance for high-demand items. The results demonstrate that RL can outperform traditional heuristics in multi-objective warehouse optimization by learning non-linear trade-offs that are difficult to encode manually.**

***Index Terms*—Item relocation, picking, reinforcement learning, simulation, slotting, warehouse optimization, warehouse densification.**

I. Introduction

Warehouse logistics has become increasingly complex because of rising product variety, volatile demand patterns, and growing pressure for fast and reliable order fulfillment. Modern warehouses operate under tight cost constraints while being required to process large volumes of heterogeneous items with high efficiency. As operations scale, inefficiencies in internal logistics, particularly those related to storage assignment and order picking can propagate into significant operational waste.

Several studies highlight that slotting and order picking are among the most cost-intensive warehouse activities, often accounting for more than half of total operating costs. It is important to emphasize that "order picking and slotting represent a high percentage of total logistics costs; therefore, improving these activities leads to significant savings in overall performance", Duque-Jaramillo et al. [7]. This makes storage decisions a strategic lever for improving both operational efficiency and economic performance.

A critical constraint in warehouse operations is storage space. Warehouse expansion or redesign is capital-intensive and often infeasible in the short term. As a result, improving space utilization within an existing layout (commonly referred to as warehouse densification) has become an important objective. However, densification is not a purely volume capacity problem. Storage decisions must consider SKU and slot dimensions, demand variability, and accessibility constraints. It is assumed that "better solutions can result from a deeper analysis of the correct positioning of products within a warehouse based on the demand for each product", Viveros et al. [17].

At the same time, picking efficiency remains a dominant performance driver. Travel distance and handling time constitute the largest share of non-value-adding activities during order fulfillment. Prior research consistently shows that positioning high-demand products closer to strategic locations, such as entry and exit points, significantly reduces picking time. Specifically, "items with higher priority must be allocated near the entry or exit doors to minimize total operation time", Duque-Jaramillo et al. [7].

To manage this complexity, warehouses often rely on classification-based storage strategies, most notably ABC classification based on demand frequency. ABC analysis enables the prioritization of high-impact SKUs and provides a practical framework for aligning product importance with storage location accessibility. However, as emphasized in the literature, "classification alone is insufficient if storage locations are not consistently aligned with SKU priority", [17].

Given the dynamic, interdependent, and stochastic nature of warehouse operations, analytical models often struggle to capture real operational behavior with sufficient fidelity. For this reason, discrete-event simulation (DES) has become a widely adopted tool for modeling and analyzing warehouse systems. DES allows the step-by-step reproduction of warehouse operations, capturing concurrency, non-linearity, and resource interactions that are difficult to represent analytically. Discrete-event simulation is particularly well suited for logistics systems, as Matloff [**?**] explains "the events being simulated—such as inventory arrivals and retrievals—are discrete and occur at specific points in time". Simulation has therefore become a standard approach for evaluating warehouse strategies under realistic operational conditions [20].

II. Problem Statement

As logistic operations are critical to business, they can not be interrupted as it would imply significant costs. Therefore, changes to layout or material-handling equipment are expensive, risky, and disruptive. For this reason, many practical solutions focus on item relocation within an existing warehouse layout, rather than structural redesign. [7] adopt this perspective by proposing slotting optimization under fixed layout constraints, reinforcing relocation as a realistic and industrially relevant approach.

However, item relocation in large-scale warehouses is a highly complex decision problem. "Storage Location Assignment Problems (SLAP) are widely recognized as NP-hard, and exact optimization approaches often rely on restrictive assumptions such as static demand, homogeneous slots, or predefined priority rules" [17]. While heuristic and deterministic methods can provide good solutions under stable conditions, they struggle to adapt to changing demand patterns and evolving warehouse states. As noted by Mojumder and Nuruzzaman [6], "traditional rule-based approaches lack the adaptability required in high-complexity logistics environments".

This complexity motivates the use of adaptive, data-driven optimization methods. In this context, reinforcement learning (RL) offers a suitable framework for addressing the warehouse item relocation problem. RL enables an agent to learn relocation policies through interaction with an environment, guided by reward signals that reflect operational objectives such as space utilization and picking efficiency. "Unlike static optimization models, RL does not require predefined allocation rules and can continuously adapt its policy as system conditions evolve", [6].

In this study, reinforcement learning is applied to the problem of item relocation in a hypothetical warehouse layout, using synthetic data that models warehouse operations, including SKU characteristics, slot capacities, demand-based ABC classification, and an initial allocation state. The objective is to develop a model that learns relocation strategies that improve warehouse densification while maintaining or reducing picking-related travel distance, without interrupting ongoing operations.

To validate the results, the learning environment is embedded within a discrete-event simulation framework. As highlighted by [20], "simulation-based approaches provide a controlled yet realistic environment in which optimization algorithms can be evaluated against multiple performance metrics without disrupting real systems". Therefore, we have formulated the following question: How can reinforcement learning be applied to the warehouse item relocation problem to improve storage space utilization while maintaining or reducing picking-related travel distance? How does it perform in comparison to other traditional methods?

III. METHODOLOGY

To address the challenge of optimizing the trade-off between volumetric storage density and operational velocity, this study adopts a comparative simulation framework. The methodology contrasts a Reinforcement Learning (RL) agent against a deterministic heuristic baseline, facilitating a direct evaluation of adaptive decision-making versus static rule-based logic within a constrained warehouse environment.

A central tenet of this methodology is the principle of "Logic Parity." To ensure a rigorous comparison, both the learning agent and the heuristic baseline operate within an identical simulation environment. They interact with a shared physics library for geometric computations, adhere to the same hierarchical constraints, and are evaluated against a unified set of Key Performance Indicators (KPIs). This design isolates the decision-making algorithm as the sole independent variable, stripping away confounding factors such as differing physics interpretations or inconsistent scoring metrics.

The methodology is structured into three distinct phases. First, a data strategy is defined to generate a statistically representative synthetic dataset, overcoming privacy restrictions associated with industrial inventory data. Second, the optimization problem is formalized through a strict hierarchy of physical and operational constraints, referred to as the "Logic Stack." Finally, an independent evaluation framework is established to audit the resulting warehouse configurations for geometric validity and safety compliance.

A. Research Design and Data Strategy

The optimization framework operates on a synthetic dataset designed to replicate the statistical and physical characteristics of a real-world automotive inventory. To mitigate the risk of exposing confidential industrial measurements, a privacy-preserving generation pipeline was developed. This strategy produces a "statistical twin" of the source inventory, ensuring that the Reinforcement Learning (RL) agent and heuristic engine are trained on data that preserves valid distribution shapes and correlations without containing any recoverable individual records [2].

1) Privacy-Preserving Synthetic Data Generation: The data generation process relies on a non-parametric approach that utilizes aggregate summary statistics rather than raw microdata. By sampling from marginal distributions, this methodology preserves global statistical properties—such as the skewness of demand or the variance of part dimensions—while breaking the linkage to specific proprietary parts [4]. To accurately mirror the relational architecture of standard Warehouse Management Systems (WMS), the resulting synthetic data is structured into three distinct, interconnected datasets:

1) **Parts Dataset:** This dataset acts as the master record for inventory items. It contains the static physical attributes generated for each unique SKU, including dimensions (Length, Width, Depth in mm), unit weight (kg), and demand profiles. Crucially, this dataset defines the intrinsic properties of the object independent of its storage status.
2) **Locations Dataset:** This dataset defines the physical infrastructure of the simulated warehouse. It maps every unique location identifier (`loc_inst_code`) to its precise 3D spatial coordinates (X, Y, Z) and defines the bounding box dimensions (width, depth, height) of the bin. This structure establishes the spatial constraints for the optimization environment.
3) **Allocations Dataset:** This dataset represents the dynamic inventory state. It links specific items from the

Parts dataset to bins in the Locations dataset. Beyond simple assignment, it captures complex stacking metadata, including the orientation of stored units (e.g., X, Y, Z alignment), the number of full versus partial layers, and the calculated volumetric utilization percentage.

This relational structure allows the simulation to distinguish between the immutable properties of an item (Parts data) and its mutable state within the warehouse (Allocations data), a distinction essential for calculating relocation costs and storage efficiency.

2) Statistical Profiling and Discretization: To ensure the synthetic data captures the non-linear physical relationships inherent in automotive parts, the generative model employs a conditional profiling technique. Relying solely on global statistics (such as mean vectors and global covariance matrices) carries a significant risk of generating physical anomalies. For instance, a global correlation model might correctly suggest that weight generally increases with dimensions, but fail to capture specific geometric aspect ratios. This could result in the generation of physically implausible items, such as a part with a length of several meters but negligible width (e.g., a structurally impossible beam) or a microscopic component with a disproportionately high mass (e.g., a 50 kg fastener).

To mitigate these artifacts and ensure that only meaningful, domain-accurate parts are generated, the methodology adopts the histogram-based conditional modeling framework proposed by Ping et al. [4]. Following this approach, the primary independent variable, *Length*, was discretized into frequency-based bins. Within each length bin, conditional descriptive statistics—specifically the 10th, 50th, and 90th percentiles—were computed for the dependent variables *Width*, *Depth*, and *Weight*. This binning strategy isolates distinct morphological categories; for example, it ensures that the weight distribution for a long, thin body panel is modeled distinctly from that of a compact, dense engine component. By retaining the bin counts from the source data for probability-weighted sampling, the model ensures the synthetic inventory mirrors the structural diversity and physical logic of the original real-world stock.

3) Constraint-Based Sampling and Dimensional Consistency: To prevent the generation of geometrically unrealistic objects (e.g., items with impossible aspect ratios or negligible density), the probabilistic sampling is augmented with deterministic physical constraints [5]. The generation pipeline adheres to a strict “Rigid Body” logic through the following steps:

1) **Hierarchical Sampling:** A length bin is selected based on source frequency, followed by uniform sampling of width and depth within the calculated inter-quantile ranges for that specific bin.
2) **Ratio-Based Correction:** The sampled dimensions are validated against plausible Width/Length and Depth/Width ratios derived from the source data. Values falling outside these bounds are clipped to the nearest valid quantile.
3) **Volumetric Consistency:** Weight is generated based on a Weight-to-Volume ratio specific to the bin, ensuring that heavy items (> 15 kg) are correlated with appropriate volumetric profiles.

These constraints ensure that the resulting dataset is not merely statistically similar but physically functional for rigid-body simulation, preventing logical errors during the geometric packing phase of the optimization.

4) Statistical Validation and Comparative Analysis: To ensure that the synthetic dataset serves as a reliable proxy for the confidential source material, a rigorous comparative validation pipeline was implemented. This framework assesses the quality of the generated data not merely through visual inspection, but through quantifiable statistical metrics, ensuring that the Reinforcement Learning agent interacts with an environment that accurately reflects the complexity and variance of the real-world operation.

The validation procedure follows a four-stage assessment protocol backed by established synthetic data evaluation frameworks:

1) **Data Harmonization and Alignment:** Prior to analysis, the confidential source dataset was transformed to match the schema of the synthetic output. This involved normalizing measurement units (converting all dimensions to millimeters and weight to kilograms) and aligning attribute semantics to ensure a direct, column-by-column comparison of physical properties.
2) **Univariate Distributional Fidelity:** The marginal distributions of all numerical attributes were evaluated to verify that the generation process reproduced the shape and spread of the original data. This involved a side-by-side comparison of descriptive statistics (mean, standard deviation, and quantiles) as recommended by Reiter [3] for assessing statistical utility. Furthermore, Probability Density Functions (PDF) and boxplots were overlaid to detect discrepancies in skewness and kurtosis, ensuring that the synthetic model captures the non-Gaussian nature of automotive inventory (e.g., the “long tail” of slow-moving parts).
3) **Multivariate Structure Preservation:** To confirm that the internal physical logic of the inventory was maintained (e.g., the relationship between volume and weight), pairwise Pearson correlation matrices were computed for both datasets. A difference matrix ($M_{diff} = |C_{real} - C_{synth}|$) was generated to quantify the divergence in variable dependencies. This metric provides a direct quantitative assessment of how well the synthetic generation process preserves the complex, high-dimensional inter-variable relationships present in the original stock.
4) **Derived Feature and Outlier Analysis:** The final validation step assessed the structural realism of the parts. Derived features—specifically geometric ratios such as density and aspect ratio—were calculated and compared. Additionally, an outlier analysis using the Interquartile Range (IQR) method was conducted to verify that the synthetic engine generated a realistic frequency of extreme values (e.g., exceptionally heavy or large parts) without introducing mathematical artifacts that could destabilize the physics engine.

B. Problem Formulation and Optimization Criteria

The optimization problem is defined as a multi-objective task: maximizing volumetric fill rates while minimizing movement costs and travel distances for high-velocity items. This dual-objective focus reflects the primary challenges in AI-driven warehouse management identified by Mojumder and Nuruzzaman [6]. To ensure the solution is robust to varying demand profiles, the system utilizes dynamic segmentation and geometric zoning, building upon the adaptive reinforcement learning frameworks established in recent studies [**?**], [8].

1) Dynamic ABC Analysis and Inventory Segmentation: In traditional warehousing, ABC classification is often based on fixed demand thresholds. However, to ensure the algorithm remains adaptable to fluctuating SKU demand—a requirement emphasized by Waubert de Puiseau et al. [8]—this study implements a dynamic percentile-based segmentation. Following the prioritization methodology established by Duque-Jaramillo et al. [7], inventory is analyzed and categorized as follows:

- **Class A (Fast Movers):** The top 20% of items by demand frequency.
- **Class B (Medium Movers):** The subsequent 30% of items.
- **Class C (Slow Movers):** The remaining 50% of items.

[Note: While Duque-Jaramillo et al. [7] and Leon et al. [14] confirm the use of ABC classification for prioritization, the specific 20/30/50 percentile split is a standard implementation used to operationalize these theories.]

Additionally, a physical safety flag is derived from item properties. Items exceeding a weight threshold ($W > 15$ kg) are designated as "Heavy Items." This aligns with the human-centric operational requirements advocated by Arslan [12], where manual picking constraints are integrated into digital twin models to ensure operator safety.

2) Warehouse Zoning and Spatial Constraints: The warehouse is modeled as a grid-based coordinate system, which is a standard representation for training agents in modern reinforcement learning environments [**?**]. To guide the optimization engine, specific sub-regions are defined based on accessibility:

- **Entrance Point:** Located at $X = 0$, serving as the reference for all velocity calculations. As demonstrated by Duque-Jaramillo et al. [7], proximity to entry/exit points is the critical factor in reducing total travel time.
- **Fast Zone:** Defined as the spatial region closest to the loading bay ($X \leq 0.25 \times X_{max}$). This zoning strategy mirrors the multi-zone layouts used for dynamic storage assignment in established RL research [8].
- **Ergonomic Zone:** A vertical band (700 mm $\leq Z \leq$ 1500 mm) optimized for human operators, as necessitated by the manual handling constraints discussed by Arslan [12].

The intersection of the Fast Zone and the Ergonomic Zone is designated as the "Target Zone" for Class A items to maximize retrieval throughput [13].

3) The Slotting Logic Hierarchy (Prioritization Stack): A core component of the methodology is the "Logic Stack," a strict hierarchical framework that governs decision-making. This sequential approach is inspired by the multi-constrained task allocation models of Oliveira et al. [16] and the sub-problem decomposition used by Viveros et al. [17].

1) **Geometric Feasibility (Hard Constraint):** An item must physically fit within a bin's dimensions. Viveros et al. [17] identify volumetric capacity and slot constraints as the foundational requirements for any slotting optimization.
2) **Safety (Hard Constraint):** Heavy items are restricted from placement above the Ergonomic Zone, reflecting the multi-constrained safety focus of contemporary warehouse models [12], [16].
3) **Macro-Placement (Efficiency):** The primary soft constraint prioritizes placing Class A items into the Target Zone, grounding the policy in frequency-based allocation logic [15].
4) **Storage Utilization (Density):** The system seeks to maximize the volumetric fill ratio. AI-driven configurations have been shown to improve space utilization by up to 35% compared to traditional methods [6].
5) **Pick Velocity (Micro-Placement):** Among locations with equal utilization potential, the bin with the lowest Manhattan distance to the entrance is selected, adhering to the travel-time metrics utilized by Duque-Jaramillo et al. [7].

C. Evaluation Framework

1) The Logic Parity Principle: To ensure a rigorous academic comparison, the study adheres to the "Logic Parity" principle. This methodology is consistent with established research protocols where Reinforcement Learning agents are benchmarked against heuristic or exact solutions within a unified simulation environment to isolate algorithmic performance from modeling discrepancies [10], [13]. By utilizing a shared physics engine and a common Logic Stack, the framework ensures that any observed improvements are attributable to adaptive decision-making logic rather than variations in environmental constraints—a control measure emphasized in the development of hybrid simulation-learning architectures [14].

2) Selection of Key Performance Indicators (KPIs): The performance of the resulting warehouse configurations is assessed using four primary metrics selected for their ability to quantify the multi-objective trade-off between spatial density and operational throughput [6], [7].

- **Volumetric Utilization (%):** Serving as the primary indicator of storage density, this metric measures the algorithm's ability to maximize volumetric capacity. Mojumder and Nuruzzaman [6] identify space utilization as a critical KPI, noting that AI-driven optimizations can yield significant gains over traditional layout methods.
- **Zone Compliance:** This metric tracks the placement of high-velocity items within their designated Target Zones. Tracking such zone-based adherence is a proven method for evaluating the effectiveness of dynamic storage location assignment, particularly when adapting to fluctuating demand profiles [8].

- **Safety Violations:** Evaluated as a binary constraint for heavy items, this metric ensures the optimization does not compromise human safety. The inclusion of such constraints reflects the human-centric requirements of modern warehouse digital twins, which must prioritize ergonomic limits to remain operationally viable [12], [16].
- **Pick Efficiency:** Measured as the aggregated Manhattan distance for Class A inventory, this metric serves as a proxy for retrieval time and labor cost. This formulation follows the methodology of Duque-Jaramillo et al. [7], who demonstrate that distance-based evaluation is the standard for assessing the efficiency of class-based slotting strategies.

IV. Modeling and Implementation

A. System Architecture and Shared Libraries

To ensure a rigorous and unbiased comparison between the Reinforcement Learning (RL) agent and the Heuristic baseline, the system architecture decouples the high-level decision-making logic from the low-level operational and analytical computations. Rather than allowing each engine to manage its own state or scoring criteria, the system interfaces with three specialized, shared libraries. This separation of concerns ensures that both algorithms operate under identical physical laws, are reported using unified visualization standards, and are subject to the same forensic audit.

The architecture is defined by three distinct support modules:

1) **The Geometric Computation Library (Physics Engine):** This module is the operational backbone of the simulation. It handles all spatial reasoning, including the calculation of 3D bin packing permutations (e.g., testing $L \times W \times D$ vs. $W \times L \times D$ orientations), collision detection, and Manhattan distance computations. Both optimization engines utilize this library for every potential move, ensuring that the fundamental question—“Does item A fit in bin B?”—is answered deterministically.
2) **The Metrics and Visualization Library (Reporting Engine):** This module standardizes the output of the experiment. It calculates Key Performance Indicators (KPIs) such as volumetric utilization, zone compliance counts, and aggregated pick distances using a consistent set of formulas. Furthermore, it manages the rendering of top-down heatmaps, front-elevation stacking plots, and demand distribution histograms. This ensures that visual comparisons between the RL and Heuristic solutions reflect actual layout differences rather than discrepancies in plotting scales or color mapping.
3) **The Independent Validation Library (Forensic Auditor):** While the first two libraries function during runtime, this module operates as a strict, post-process auditor. As detailed in Section IV-F, it performs a “black-box” verification of the final output files. It re-calculates all volumes from raw dimension data to ensure conservation of mass, checks for rigid-body violations, and distinguishes between physical capacity limits and algorithmic failures.

By centralizing these critical functions, the architecture enforces the “Logic Parity” principle, isolating the decision-making strategy as the sole independent variable in the investigation.

B. Geometric Computation and Physics Engine

1) **Dimensional Feasibility and Orientation Handling:** Locations and SKUs are represented as three-dimensional cuboids with dimensions in millimeters. For a given SKU–location pair, the engine evaluates all possible orthogonal orientations of the SKU by permuting its length, width, and depth, resulting in 6 possible orientations for a given SKU. For each orientation, the algorithm computes how many units can be placed along each axis of the storage slot using integer division, ensuring that no dimensional overlap or boundary violation occurs.
   This process results in a three-dimensional grid defined by (nx,ny,nz), representing the maximum number of units that can be placed along each axis. The orientation that yields the highest number of feasible units is selected as the best orientation, and its corresponding grid defines the maximum storage capacity of that SKU in the given location. If no orientation produces a positive capacity, the SKU is deemed physically infeasible for that location (since no instance of this SKU can be allocated).
2) **Layered Packing:** Beyond capacity estimation, the engine explicitly models how items occupy space within a location. Given an initial number of allocated units, the grid is decomposed into fully filled horizontal layers (in x and y axes) and, if necessary, a partially filled final layer. Represents how products are stacked within the storage slot. For diagnostic and validation purposes, the engine can generate explicit two-dimensional matrices representing the occupancy of each layer. Although not required for optimization, this capability provides an additional level of transparency and supports visual inspection of packing patterns during debugging and analysis.
3) **Initial Allocation:** The Geometric Computation Engine is also used during the construction of the initial warehouse state. At simulation time t=0, the inventory is allocated in a random manner to avoid introducing structural bias. However, this randomness is strictly bound by geometric feasibility. Moreover, this algorithm processes SKUs in descending order of geometric difficulty, defined by their largest dimension, in other words, high volume SKUs are allocated first to avoid fragmentation of storage capacity and increases the likelihood that large items find feasible locations. For each SKU, the algorithm iteratively searches for empty locations where a feasible allocation grid exists. When a suitable location is found, the location is filled up to its maximum feasible capacity or until the available

stock is exhausted. This procedure distinguishes between different causes of allocation failure. SKUs may remain unallocated because they do not physically fit in any warehouse location, because no free locations remain, or because feasible locations existed but were exhausted earlier.

4) **Allocation Grid as a Shared Physical Constraint:** The allocation grid computed by the geometry engine is treated as a hard physical constraint throughout the simulation. Both the RL agent and the heuristic engine must respect the same grid limits, orientation choices, and volumetric capacities when proposing relocation actions. No algorithm is allowed to override or approximate these constraints. This design ensures that differences in performance between optimization methods arise exclusively from decision-making quality rather than from inconsistent feasibility assumptions. If an allocation or relocation is accepted by the system, it is guaranteed to be physically realizable under the modeled warehouse conditions.
5) **Integration with Distance and System-Level Modeling:** This engine is complemented by a distance computation module that evaluates Manhattan distances between storage locations based on their spatial coordinates. This enables higher-level modules to assess accessibility and travel cost without embedding geometric logic into decision-making algorithms. The overall simulation flow is orchestrated by a central warehouse controller, which sequentially loads data, constructs the initial geometry-feasible allocation, and exposes the resulting warehouse state to downstream simulation and optimization components. This layered architecture ensures a clear separation between physical laws, system behavior and optimization logic.

C. The Metrics and Visualization Library

While the Geometric Library handles the physical feasibility of individual moves, the Metrics and Visualization Library is responsible for the holistic evaluation of the warehouse state. This module standardizes the calculation of Key Performance Indicators (KPIs) and generates visual diagnostics to verify that the optimization algorithms are not merely exploiting mathematical loopholes but are producing operationally viable layouts.

a) Standardized Performance Metrics: To assess the trade-off between density and accessibility, the library computes four primary metrics derived from the reviewed literature:

1) **Volumetric Utilization Rate (U_{vol}):** Defined as the ratio of the total volume of stored SKUs to the total volume of allocated bin space:

$$U_{vol} = \frac{\sum_{i=1}^{n}(V_{item,i} \times Q_{item,i})}{\sum_{j=1}^{m} V_{bin,j}} \times 100$$

This metric serves as the primary indicator of storage density. As noted in a recent systematic review by Mojumder and Nuruzzaman [6], AI-driven layout optimization is expected to yield utilization improvements in the range of 20–35% over static baselines.
2) **Operational Velocity Score (Pick Efficiency):** This metric aggregates the retrieval cost for high-demand items, calculated as the sum of Manhattan distances (L_1 norm) from the warehouse entrance point for all Class A items, weighted by their demand frequency. This formulation aligns with the approach by Duque-Jaramillo et al. [7], who demonstrate that prioritizing high-demand SKUs through class-based slotting is a critical factor in minimizing travel distances and overall warehouse operational costs.
3) **Zone Compliance Index:** A count-based metric that verifies adherence to the "Golden Zone" logic, calculated as the percentage of Class A items successfully placed within the designated *Target Zone* (the intersection of high-velocity and ergonomic access). As demonstrated by Waubert de Puiseau et al. [8], deep reinforcement learning architectures are particularly effective at solving dynamic storage location assignment problems by learning to adapt zone-based placements to seasonal demand fluctuations and evolving order profiles.
4) **Safety Constraint Validation:** A binary metric used to flag critical failures. It checks for the presence of items flagged as "Heavy" (> 15 kg) located above the ergonomic threshold ($Z > 1500$ mm). Any value greater than zero indicates an invalid configuration.

b) Visualization Pipeline: To facilitate the qualitative analysis of the layout logic, the library includes a rendering engine capable of generating orthogonal projections of the warehouse state.

- **Top-Down Heatmap ($X - Y$ Plane):** This visualization maps the warehouse floor, coloring each bin stack based on its aggregate volumetric utilization. This view allows for the rapid identification of "cold zones" (under-utilized space) and verifies that the algorithm is densifying the layout uniformly rather than creating isolated clusters.
- **Front Elevation Composite ($X - Z$ Plane):** To provide a comprehensive analysis of vertical slotting across the entire facility, this plot flattens the 3D warehouse depth into a single 2D panoramic view. It conceptually "unfolds" the warehouse, projecting parallel aisles side-by-side along the horizontal axis.
  In this view, every storage location is represented as an individual cell at its specific vertical level. The visualization encodes **Volumetric Utilization** through two simultaneous visual channels:
  1) **Geometric Fill:** The shaded area within each cell corresponds literally to the fill percentage (e.g., a bin that is 50% full appears half-filled).
  2) **Color Coding:** A chromatic gradient is applied to the fill area, providing an immediate heat-map indication of density (e.g., distinguishing completely empty bins from fully saturated ones).

  This composite view is critical for verifying that the "Logic Stack" is correctly utilizing the Ergonomic Zone (700–1500 mm) without violating vertical safety constraints.

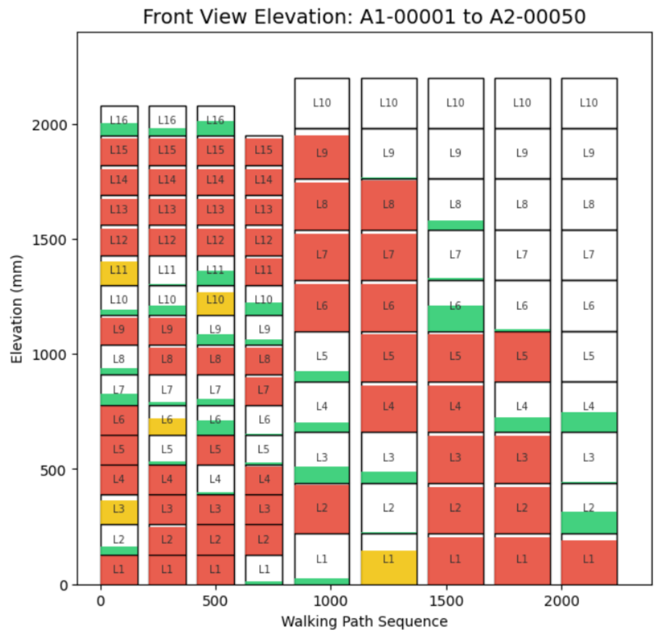

Fig. 1: **Front Elevation Composite Visualization.** This diagnostic plot aggregates multiple warehouse aisles into a continuous side-by-side projection to visualize vertical distribution. The vertical axis represents the rack levels. Each cell represents a specific bin, where both the fill height and color intensity indicate the **Volumetric Utilization %**. This allows for visual confirmation that high-density and heavy items are correctly settled into lower and middle tiers.

D. Heuristic Baseline Engine

The heuristic baseline serves as a deterministic optimization policy used to establish a high-performance reference for warehouse slotting. The engine transitions the warehouse from an initially random allocation state to a structured, class-based configuration by maximizing a multi-objective scoring function while strictly enforcing physical and safety constraints. The overall design follows established goods-location optimization approaches in the literature, where "greedy, priority-driven relocation strategies are applied to reduce picking cost by sequentially assigning high-turnover items to more favorable locations under fixed layout constraints", as [18] explains.

a) System Inputs and Data Pre-processing: The optimization framework operates on three primary input datasets:

1) **Warehouse Topology:** A discrete representation of 357 storage bins defining spatial coordinates (x, y, z) and physical dimensions (Width, Depth, Height).
2) **Item Dataset:** A catalog of 174 slots were occupied, each slot contains only one type of SKU, complying with physical dimensions, weight, and historical demand information, generated synthetically.
3) **Initial Allocation State:** It is the current mapping of inventory items to storage bins.

During pre-processing, the engine performs a demand-driven **ABC analysis**. Items are ranked by demand frequency and classified as follows: the top 20% are labeled **Class A** (high priority), the next 30% as **Class B**, and the remaining items as **Class C**. This classification determines the processing order of the heuristic, ensuring that high-impact items are optimized first.

b) Heuristic Optimization Algorithm: The engine applies a *Greedy Strategic Relocation* algorithm. Each allocated item is evaluated sequentially against all currently empty bins to identify a superior placement. A relocation is executed only if the candidate bin improves the total score by more than a predefined threshold, $\Delta S > 5$.

The suitability of assigning item i to bin b is quantified by the objective function:

$$S(i, b) = R_{\text{zone}}(i, b) + R_{\text{util}}(i, b) - P_{\text{dist}}(b)$$

- **Zone Reward R_{zone}:** A binary incentive mechanism promoting strategic placement.
  - **Fast Zone:** +500 points if a Class A item is placed within the front 25% of the warehouse, defined by $X \leq 0.25X_{\max}$.
  - **Ergonomic Zone:** +500 points if a heavy item (> 15 kg) is placed within the ergonomic "Golden Zone" (700–1500 mm).
- **Utilization Reward R_{util}:** A continuous score in the range $[0, 1000]$ reflecting volumetric efficiency:

$$R_{\text{util}} = \frac{Q \cdot V_{\text{unit}}}{V_{\text{bin}}} \times 1000$$

- **Distance Penalty P_{dist}:** A cost proportional to the Manhattan distance from the warehouse entrance $(0, Y_{\text{mid}})$:

$$P_{\text{dist}} = \frac{d_{\text{entrance}}}{d_{\max}} \times 100$$

c) Geometric and Safety Constraints: To ensure physical feasibility, the heuristic integrates a **6-axis geometric validation** module implemented in `geometry.py`. For each candidate bin, all six permutations of the item dimensions (e.g., $L \times W \times D$, $W \times L \times D$) are evaluated.

A bin is considered admissible only if all of the following conditions are satisfied:

1) **Fit Constraint:** The computed item grid under a valid orientation fits entirely within the bin boundaries.
2) **Safety Constraint:** Items exceeding 15 kg are strictly prohibited from being placed above a height of 1.5 m to prevent structural instability and damage to lower layers.

E. Reinforcement Learning (RL) Implementation

The warehouse relocation problem is formulated as a finite-horizon, episodic Markov Decision Process (MDP), in which the placement of inventory is modeled as a sequence of discrete, box-level decisions. At each decision step, the agent selects a single storage location for one box, observes the immediate outcome of this action, and proceeds to the next placement until all available inventory has been allocated or no feasible placements remain. This sequential decision-making formulation follows the reinforcement learning abstractions commonly adopted in warehouse logistics optimization [**?**] and operationalized in dynamic storage assignment [8].

The MDP formulation is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, P the (implicit) state transition dynamics, and R the reward function. The transition dynamics are deterministic with respect to the physical feasibility checks and inventory updates, as all geometric validation and capacity constraints are resolved by an external physics-based allocation engine shared with the heuristic baseline. Consequently, uncertainty in the environment arises not from stochastic transitions, but from the evolving occupancy of the warehouse as sequential placement decisions are executed. This deterministic transition modeling for storage location assignment is consistent with the frameworks proposed by Troch et al. [15] and Liu et al. [10].

Each episode corresponds to a sequence of independent placement decisions sampled over randomly generated warehouse occupancy snapshots during training. The agent does not attempt to model long-term rollouts of future placements explicitly. Instead, a one-step temporal difference update is employed, resulting in a contextual bandit–style learning regime. This design choice reflects the industrial requirement for strict constraint enforcement and interpretability [9], while still allowing the agent to learn statistically grounded preferences over placement locations under varying congestion conditions. Similar hybrid simulation–RL strategies have been successfully applied in warehouse operations by Leon et al. [14].

Importantly, the RL agent does not replace the deterministic feasibility logic. All actions—both during training and optimization—are evaluated through the same geometric computation, distance metrics, and safety constraints as the heuristic engine. The role of reinforcement learning is therefore limited to learning a value function over admissible actions, enabling adaptive prioritization under dynamic warehouse states without violating predefined business or physical rules. This architectural separation aligns with safe and human-centric warehouse optimization frameworks [12], [16].

1) State Space Representation: The RL state space encodes both *item-specific attributes* and the *current warehouse occupancy*, reflecting the requirement that the agent must be aware of congestion effects and remaining capacity during sequential placement decisions. Each state is constructed at the moment a single box is to be placed and is defined as:

$$s = \big(c_i,\ v_i,\ h_i,\ o_{FE},\ o_{FnE},\ o_{nFE},\ o_{nFnE}\big)$$

where:

- $c_i \in \{A, B, C\}$ denotes the Dynamic ABC class of item i, computed prior to optimization based on demand percentiles.
- $v_i \in \{0, 1, 2\}$ represents a discretized volume bucket derived from the empirical distribution of item unit volumes (lower, medium, upper tercile).
- $h_i \in \{0, 1\}$ is a binary indicator identifying heavy items, defined as $WT\_KG > 15\,\text{kg}$.
- $o_{FE}, o_{FnE}, o_{nFE}, o_{nFnE} \in \{0, 1, 2, 3\}$ denote coarse occupancy buckets for the four warehouse macro-zones: Fast–Ergonomic (FE), Fast–NonErgonomic (FnE), NonFast–Ergonomic (nFE), and NonFast–NonErgonomic (nFnE).

Zone occupancy is expressed as the ratio of occupied bins to total bins within each zone and discretized into quartile-based buckets. This abstraction balances representational expressiveness with tractability, allowing the agent to distinguish between sparse and saturated regions of the warehouse without incurring the dimensionality of a full binary occupancy map.

This design directly addresses the "current warehouse occupancy" requirement specified in the RL formulation and enables learning of congestion-aware placement strategies rather than purely myopic, item-centric behavior.

2) Action Space and Macro-Zone Selection: In accordance with the formal RL specification, the action space consists of selecting a *specific storage location* for the placement of a single box:

$$a \in \mathcal{A} = \{\texttt{loc\_inst\_code}_1, \ldots, \texttt{loc\_inst\_code}_N\}$$

where each action corresponds to one physical bin defined in the warehouse layout dataset. This formulation preserves full compliance with the rulebook definition of the action space and avoids abstracted or indirect control representations.

Although warehouse macro-zones (Fast, Ergonomic) play a critical role in prioritization, they are *not* encoded as actions. Instead, macro-zone logic influences decision-making indirectly through:

- state variables capturing zone-level congestion, and
- reward shaping that favors high-demand items placed in Fast and Ergonomic zones.

This indirect encoding strategy is consistent with hybrid learning–rule approaches reported in smart warehouse task assignment [13], [14].

This separation ensures that the RL agent retains full spatial freedom while still internalizing operational heuristics through learning rather than hard-coded constraints. Conceptually, the agent performs micro-level bin selection while learning macro-level zoning preferences via cumulative reward optimization.

During optimization, actions that violate hard constraints (e.g., mixed storage or unsafe height placement of heavy items) are evaluated but immediately penalized, preventing their reinforcement.

3) Learning Algorithm: Tabular Q-Learning: The reinforcement learning engine is trained using *tabular Q-learning*, a value-based method that estimates the expected return of placing a box into a candidate storage bin under the current state [21], [22]. For a given state–action pair (s, a), the action-value function $Q(s, a)$ represents the expected cumulative reward obtained by executing action a in state s and following the learned policy thereafter. In the warehouse setting, s encodes the current item attributes and a coarse representation of macro-zone congestion, while a corresponds to selecting a specific storage location identifier for a *single box-level placement*.

Training proceeds over a finite number of episodes, each constructed from randomized warehouse occupancy snapshots. Within each episode, the algorithm applies an ϵ-greedy exploration policy: with probability ϵ, a feasible action is sampled uniformly (exploration), and with probability $1-\epsilon$, the action with the highest estimated value is selected (exploitation). This strategy is commonly used in discrete action RL to balance exploration and exploitation without requiring model knowledge of the transition kernel [21]. Infeasible actions are not executed; instead, they are effectively *masked* by the physics-based feasibility engine (geometry and safety constraints), or they are assigned the hard penalty $R = -10{,}000$ as defined in the reward structure. This ensures that learning remains grounded in physically admissible placements and that invalid placements cannot be reinforced.

The Q-learning update rule is applied after each placement attempt. Let r_t denote the immediate reward obtained after placing one box, and let s_{t+1} be the updated state after the warehouse occupancy map is deterministically updated. The tabular update is given by:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big(r_t + \gamma \max_{a' \in \mathcal{A}(s_{t+1})} Q(s_{t+1}, a') - Q(s_t, a_t)\Big) \tag{1}$$

where $\alpha \in (0, 1]$ is the learning rate controlling the update magnitude, $\gamma \in [0, 1]$ is the discount factor, and $\mathcal{A}(s_{t+1})$ denotes the admissible action set after feasibility filtering. In the implemented training regime, the learning problem is intentionally *short-horizon* due to strict feasibility constraints and the focus on immediate placement quality; consequently, γ is set to a small value or 0 to reflect a contextual bandit–style approximation [9], [21]. The exploration rate ϵ is initialized to encourage broad coverage of candidate bins and decayed across episodes to stabilize on higher-value placements as training progresses:

$$\epsilon \leftarrow \max(\epsilon_{\min}, \epsilon \cdot \lambda), \tag{2}$$

with decay factor $\lambda \in (0, 1)$ and lower bound $\epsilon_{\min}$.

The resulting Q-table is then used during optimization to rank candidate bins for each sequential box placement. By construction, the policy is constrained to preserve single-SKU-per-bin storage, enforce heavy-item height safety, and rely exclusively on the shared geometry solver for fit and capacity determination. This yields a learning-driven prioritization mechanism embedded within a deterministic, rule-compliant relocation pipeline, consistent with the safety and deployment considerations emphasized in real-world reinforcement learning [9].

4) Reward Function Design and Penalty Structure: The reward function is defined to be numerically identical to the scoring logic used by the heuristic engine, ensuring logic parity between both optimization approaches. Each action results in an immediate scalar reward computed after attempting to place exactly one box.

a) Hard Penalties (Illegal Moves): Any action violating a non-negotiable constraint yields a fixed penalty:

$$R = -10{,}000$$

This occurs if:

- the external allocation library reports geometric infeasibility, or
- a heavy item ($WT\_KG > 15\,\text{kg}$) is placed above $1500\,\text{mm}$ in the vertical axis.

These penalties effectively terminate learning for unsafe or physically impossible actions by overwhelming any potential positive reward.

b) Scaled Rewards (Legal Moves): If the placement is feasible, the total reward is computed as:

$$R = R_{\text{zone}} + R_{\text{util}} + R_{\text{dist}} + R_{\text{aff}}$$

- **Zone Reward R_{zone}:**
$$\begin{cases} +1000, & \text{Class A item in Fast \& Ergonomic zone} \\ +400, & \text{Class B item in Fast \& Ergonomic zone} \\ 0, & \text{otherwise} \end{cases}$$
- **Utilization Reward R_{util}:**
$$\left(\frac{Q \cdot V_{\text{unit}}}{V_{\text{bin}}}\right) \times 800$$
where Q is the number of boxes stored in the bin after placement.
- **Distance Penalty R_{dist}:**
$$\left(\frac{d_{\text{entrance}}}{d_{\max}}\right) \times (-100)$$
using Manhattan distance in the horizontal plane only.
- **Affinity Reward R_{aff}:**
$+50$ if a neighboring bin contains an item with $\pm 15\%$ similar unit volume.

The reward structure reflects a lexicographic operational priority: safety and feasibility dominate all decisions, followed by long-term storage density, picking efficiency, and organizational coherence. Although individual placements into large bins may initially appear suboptimal in terms of utilization percentage, the cumulative reward formulation favors consolidation over time, aligning with the global capacity optimization objective of the system.

5) Training Configuration and Hyperparameters: The reinforcement learning agent is trained using a tabular value-based learning approach with one-step temporal difference updates, reflecting the contextual bandit nature of the placement decision problem. The learning configuration is intentionally conservative to ensure stability under hard physical constraints and deterministic environment transitions.

The primary training hyperparameters are defined as follows:

- **Learning rate ($\alpha = 0.25$):** A moderately high learning rate is selected to enable rapid adaptation under deterministic transitions, while avoiding instability caused by large reward magnitudes.
- **Discount factor ($\gamma = 0.0$):** The discount factor is set to zero because rewards are computed immediately after each placement decision. No long-horizon credit assignment is required, aligning the learning process with a contextual bandit setting.
- **Exploration strategy:** An ϵ-greedy policy with a fixed or slowly decaying exploration rate is employed, ensuring sufficient exploration of feasible storage locations while

gradually favoring higher-valued actions.

- **Exploration rate (ϵ):** Initialized at $\epsilon = 1.0$ and linearly decayed to $\epsilon = 0.05$ over training. This ensures broad exploration in early episodes and increasingly stable decision-making as learning progresses.
- **Training episodes ($N = 6000$):** Several thousand independent episodes are executed, each starting from randomized warehouse occupancy states, to promote generalization and prevent overfitting to specific layouts.

In addition to standard RL hyperparameters, several environment-level knobs directly influence learning behavior:

- **Randomized initial occupancy:** Each episode begins from a randomly generated warehouse state derived from the synthetic data generator, preventing overfitting to a single layout.
- **Action masking:** Infeasible actions are dynamically filtered or heavily penalized using the external feasibility engine, reducing the effective action space.
- **Reward scaling constants:** Fixed coefficients in the reward function (e.g., utilization and distance multipliers) act as implicit policy-shaping parameters and were tuned to ensure balanced trade-offs between space efficiency and operational accessibility.

This training configuration prioritizes robustness, interpretability, and safe deployment over aggressive convergence, aligning with best practices for real-world reinforcement learning systems operating under strict physical and business constraints [9], [**?**].

F. Independent Validation and Auditing

To ensure the reliability of the optimization results, the research framework employs an Independent Validation Library. This module operates as a strict, post-process auditor, functionally separated from the optimization engines. This design aligns with the principles of "Operational Validity" defined by Sargent [1], utilizing what is effectively a black-box testing approach to verify that the simulation's output behavior remains physically accurate and consistent, regardless of the decision-making algorithm employed.

While the RL agent and Heuristic engine calculate scores during runtime to guide decision-making, this library performs a forensic analysis on the final configurations. Its primary function is to certify that the proposed warehouse states are physically valid and compliant with safety constraints. This is a critical requirement for real-world RL applications, where, as identified by Dulac-Arnold et al. [9], agents may otherwise exploit simulation artifacts or underspecified reward functions to maximize returns at the expense of physical safety or logical consistency.

The auditing process is structured into three critical verification layers:

1) Data Conservation and Rigid Body Integrity: The first layer of verification ensures that the fundamental data properties remain invariant throughout the simulation, adhering to standard model verification techniques.

- **Inventory Conservation:** The auditor enforces a strict "Conservation of Mass" principle. It verifies that the sum of allocated items plus unallocated items exactly equals the initial demand input. This prevents algorithmic errors where items might be inadvertently duplicated or deleted during state transitions.
- **Rigid Body Verification:** To prevent mathematical artifacts, the library compares the dimensions of every allocated item against the master *Parts* dataset. This ensures that the optimization engines did not mathematically "compress" or morph items to fit into smaller bins—a common issue in numerical packing algorithms.
- **Volumetric Accuracy:** The utilization percentages are re-calculated from raw dimensions $(L \times W \times D)$ rather than relying on the cached values reported by the agents.

2) Geometric Compliance and Constraint Enforcement: The second layer audits the spatial validity of the placement against the hard constraints of the Logic Stack. As noted by Dulac-Arnold et al. [9], enforcing constraints via a separate safety layer is often necessary in industrial RL to prevent "negative side effects" such as unsafe stacking.

- **Stack Feasibility:** The auditor validates that the specific grid layout (rows $\times$ columns $\times$ layers) proposed by the agent physically fits within the bin boundaries.
- **Collision Detection:** A global coordinate check is performed to ensure no two items occupy the same 3D voxel space (physical overlap).
- **Rule Enforcement:** The configuration is scanned for violations of business rules, specifically the "Single SKU per Bin" constraint and the safety prohibition against placing heavy items ($>$ 15 kg) above the ergonomic height threshold ($Z > 1500$ mm).

3) Feasibility Analysis of Unallocated Inventory: The final layer addresses the system's failure modes. In scenarios where the algorithm fails to place the entire inventory, the auditor distinguishes between physical saturation and algorithmic inefficiency. It runs a brute-force "Fit Test," attempting to place every unallocated item into every remaining empty bin.

- If the Fit Test fails for all items, the non-allocation is classified as a **Capacity Limitation** (the warehouse is physically full).
- If the Fit Test succeeds, the non-allocation is flagged as an **Algorithmic Failure** (the agent failed to find a valid solution that existed).

This distinction is crucial for properly evaluating the RL agent's performance, decoupling environmental limits from policy quality.

V. Results

This section presents a comprehensive evaluation of the warehouse layout optimization outcomes obtained for a prototype warehouse comprising **357 distinct storage locations** and a corresponding set of synthetic inventory items with heterogeneous physical dimensions, weights, and demand profiles. The objective of this evaluation is to assess how effectively different allocation strategies translate predefined operational objectives—namely spatial efficiency, accessibility, and prioritization—into physically realizable and operationally meaningful warehouse layouts. Particular emphasis is placed on understanding how well each approach balances competing constraints such as storage density, picking efficiency, ergonomic safety, and demand-driven prioritization within a shared physical environment.

The analysis is structured around three progressively refined warehouse states. First, the *Baseline* configuration represents the initial allocation prior to optimization, reflecting a non-prioritized placement of items without systematic enforcement of demand-based zoning, ergonomic constraints, or distance minimization. This state serves as a reference point, capturing the inherent inefficiencies and accessibility challenges present in unoptimized or legacy warehouse layouts. Second, the *Heuristic-Optimized* configuration applies a deterministic slotting logic derived from domain knowledge, including inventory segmentation and rule-based prioritization. This intermediate state demonstrates the effectiveness and limitations of static, expert-driven decision rules when applied to complex spatial allocation problems. Third, the *Reinforcement Learning (RL)-Optimized* configuration reflects the outcome of an agent trained to sequentially relocate items while maximizing a structured reward function subject to strict physical, ergonomic, and operational constraints, thereby enabling adaptive decision-making under dynamic congestion conditions.

All results are evaluated on a common warehouse geometry and inventory instance to ensure direct comparability across methods. This controlled evaluation setting ensures that observed differences in performance are attributable to the optimization logic itself rather than variations in layout or demand input. The evaluation combines spatial visualizations, statistical distributions, and quantitative performance metrics to provide a multi-faceted assessment of each approach. Spatial plots are used to qualitatively assess zoning compliance, ergonomic accessibility, and overall layout coherence, while numerical metrics provide objective measures of space utilization efficiency, constraint adherence, and aggregate placement quality.

The results section is therefore organized into three main subsections—Baseline, Heuristic, and Reinforcement Learning—each following a consistent analytical structure to facilitate comparison. For each warehouse state, the discussion begins with spatial interpretations of the layout from top-down and front-elevation perspectives, followed by an examination of demand-to-accessibility relationships and volume utilization characteristics. Each subsection concludes with a detailed presentation and interpretation of aggregated performance metrics. A final summary subsection synthesizes the comparative findings and highlights the relative strengths, trade-offs, and limitations of the evaluated optimization approaches.

A. Baseline Results (Initial Warehouse State)

The baseline represents the initial warehouse configuration prior to any optimization. Item placements correspond to the input allocation without enforcement of the full slotting hierarchy, dynamic ABC prioritization, or ergonomic and fast-zone alignment.

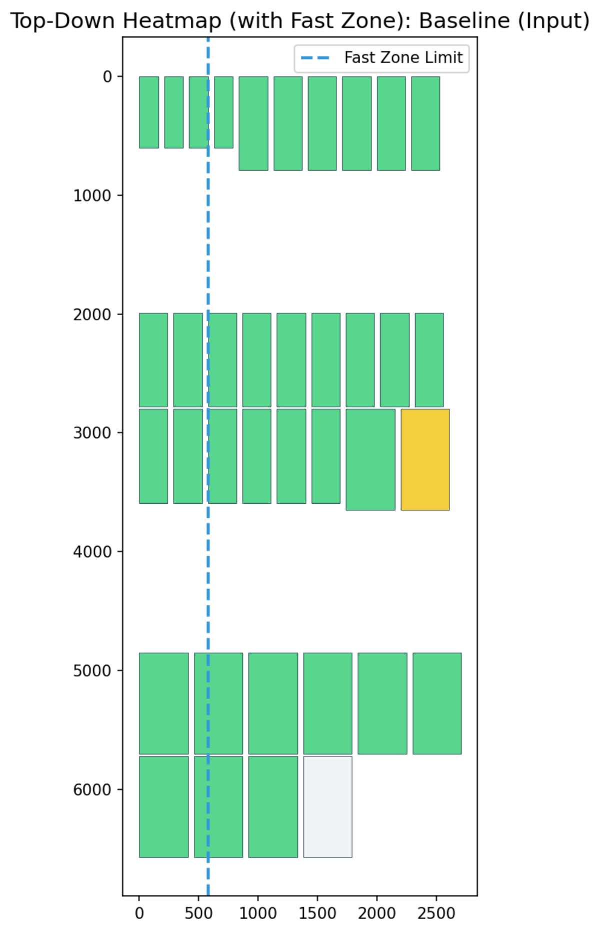

Fig. 2: Top-down warehouse layout with fast-zone boundary (Baseline)

1) Top-Down Spatial Distribution: Figure 2 illustrates the baseline top-down warehouse layout, where each rectangle represents a storage bin projected onto the floor plane. The dashed vertical line denotes the fast-zone boundary, separating locations with shorter travel distances from those farther from the warehouse entrance.

In the baseline configuration, item placements appear largely uncorrelated with the fast-zone constraint. High-demand and priority items are distributed across both near and far locations, indicating the absence of demand-aware spatial organization. This results in unnecessary travel distances for frequently accessed items, directly impacting picking efficiency.

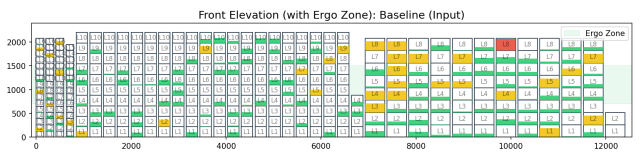

Fig. 3: Front elevation with ergonomic zone overlay (Baseline)

2) Front Elevation and Ergonomic Accessibility: Figure 3 presents the front elevation view of the warehouse, showing vertical slot levels across aisles. The shaded horizontal band indicates the ergonomic zone, corresponding to height ranges suitable for frequent manual handling.

In the baseline state, high-priority (Class A) and moderate-priority (Class B) items are frequently placed outside the ergonomic zone, including at very low or high elevations. This misalignment increases physical strain during picking operations and violates ergonomic design principles commonly recommended in warehouse layout literature.

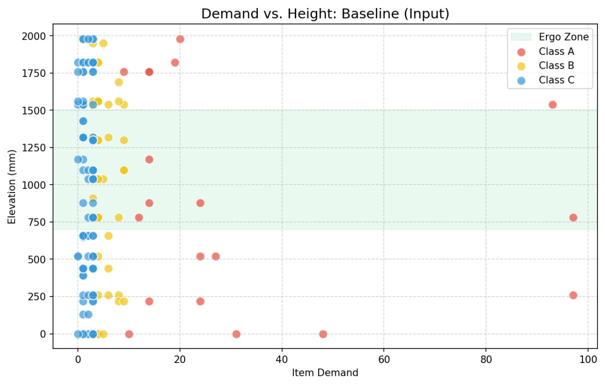

Fig. 4: Item demand versus elevation with ergonomic zone (Baseline)

3) Demand Versus Elevation Relationship: Figure 4 plots item demand against vertical placement height, color-coded by ABC class. Ideally, high-demand items should cluster within the ergonomic zone.

The baseline configuration shows no clear correlation between demand and elevation. Several high-demand Class A items are placed well outside the ergonomic band, highlighting inefficiencies in manual accessibility and increased operational effort per pick.

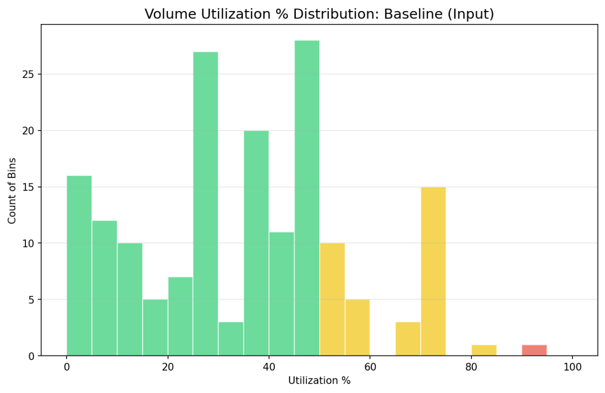

Fig. 5: Bin volume utilization distribution (Baseline)

4) Volume Utilization Distribution: Figure 5 depicts the distribution of volume utilization across occupied bins. The histogram reveals a wide spread, with a significant number of bins operating at low utilization levels.

This uneven distribution indicates inefficient space usage, where some bins are underfilled while others approach higher utilization, increasing fragmentation and reducing effective storage density.

TABLE I: Baseline Performance Metrics

Metric	**Value**
Weight Violations	0
Misplaced Class A Items	17
Average Combined Score	261.0
Zone Reward (Avg.)	27.6
Utilization Reward (Avg.)	285.2
Distance Penalty (Avg.)	-51.8
Average Utilization (Occupied Bins)	35.65%
Occupied Bin Ratio	48.7% (174/357)

5) Quantitative Performance Metrics: The baseline metrics quantify the qualitative observations from the visualizations. While no hard physical violations occur, the high number of misplaced Class A items confirms poor prioritization. The relatively low combined score reflects weak zone alignment and significant distance penalties, despite moderate utilization rewards.

B. Heuristic Optimization Results

The heuristic-optimized configuration represents the warehouse layout obtained by applying a deterministic slotting strategy derived from domain-specific rules, including dynamic ABC prioritization, fast-zone allocation, and ergonomic height constraints. The heuristic operates under the same physical feasibility checks and geometric validation logic as the baseline and RL approaches, ensuring direct comparability.

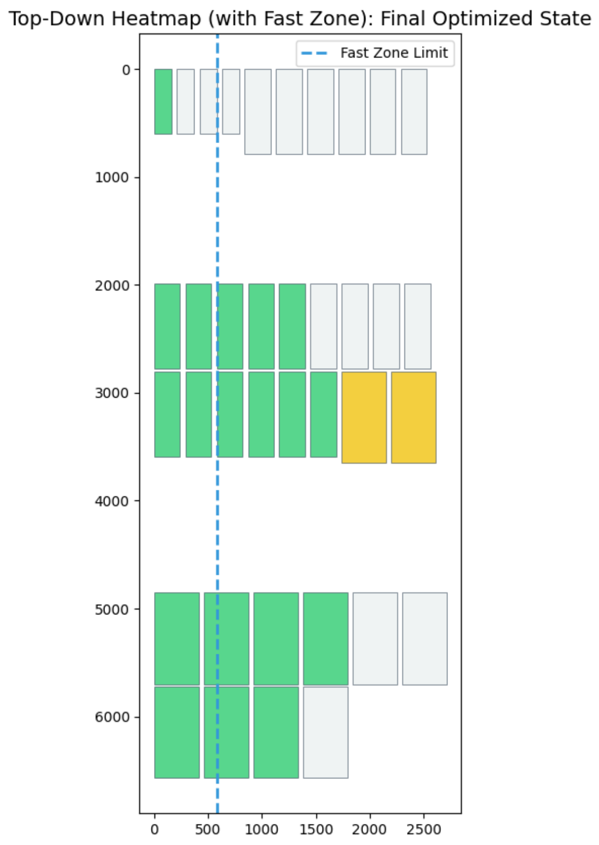

Fig. 6: Top-down warehouse layout with fast-zone boundary (Heuristic Optimized)

1) Top-Down Spatial Reorganization: Figure 6 illustrates the top-down spatial distribution after heuristic optimization. Compared to the baseline, high-demand items exhibit a stronger concentration within the fast-zone boundary, indicating effective rule-based prioritization of frequently accessed inventory. Medium-demand items are partially aligned with the fast zone but still exhibit some dispersion into non-fast regions. This residual spread reflects the static and greedy nature of the heuristic logic, which does not adapt dynamically to congestion or global trade-offs.

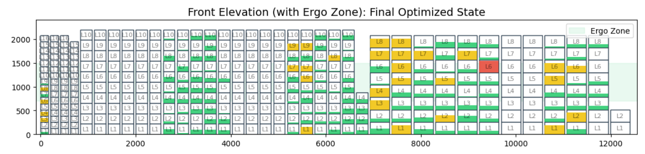

Fig. 7: Front elevation with ergonomic zone overlay (Heuristic Optimized)

2) Ergonomic Zone Compliance: As shown in Figure 7, the heuristic method substantially improves ergonomic compliance relative to the baseline. Most Class A and Class B items are repositioned within the ergonomic height band, and heavy items are constrained to safe vertical levels. Minor deviations persist in densely populated regions, where deterministic rules compete for limited ergonomic space.

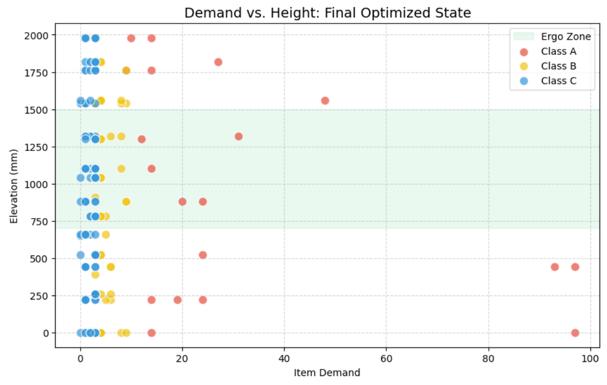

Fig. 8: Item demand versus elevation with ergonomic zone (Heuristic Optimized)

3) Demand-Aware Vertical Placement: Figure 8 demonstrates an improved relationship between item demand and vertical placement. High-demand items exhibit a stronger tendency to cluster within the ergonomic zone compared to the baseline. Nevertheless, the spread remains broader than in the RL-optimized case, highlighting the heuristic's limited ability to adapt dynamically to congestion patterns.

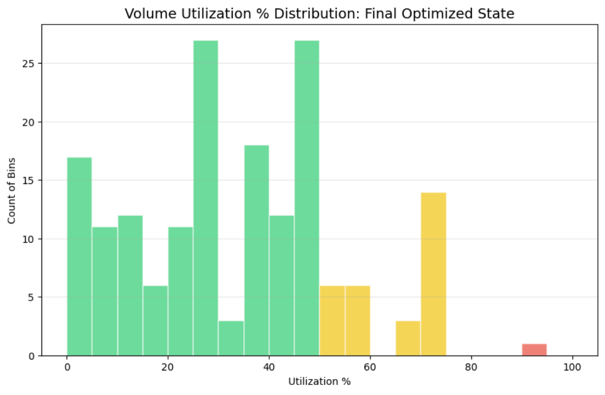

Fig. 9: Bin volume utilization distribution (Heuristic Optimized)

4) Volume Utilization Distribution: The utilization histogram in Figure 9 indicates a modest reduction in extreme underutilization compared to the baseline. While overall average utilization remains comparable, the heuristic improves spatial consistency by reducing fragmented allocations. However, it does not significantly increase consolidation efficiency due to its rule-based and non-adaptive structure.

TABLE II: Heuristic Optimized Performance Metrics

Metric	Value
Weight Violations	0
Misplaced Class A Items	14
Average Combined Score	293.6
Zone Reward (Avg.)	54.0
Utilization Reward (Avg.)	273.3
Distance Penalty (Avg.)	-33.6
Average Utilization (Occupied Bins)	34.16%
Occupied Bin Ratio	48.7% (174/357)

5) Heuristic Performance Metrics: The quantitative metrics confirm that the heuristic approach delivers a clear improvement over the baseline, particularly in zone alignment and distance reduction, while maintaining full physical feasibility. However, the persistence of misplaced Class A items and limited gains in utilization underscore the heuristic's inability to resolve complex trade-offs under high spatial congestion, motivating the use of adaptive learning-based optimization.

C. Reinforcement Learning Optimized Results

The RL-optimized results correspond to the final warehouse configuration obtained after training the agent using the guide-parity reward function and constrained action space.

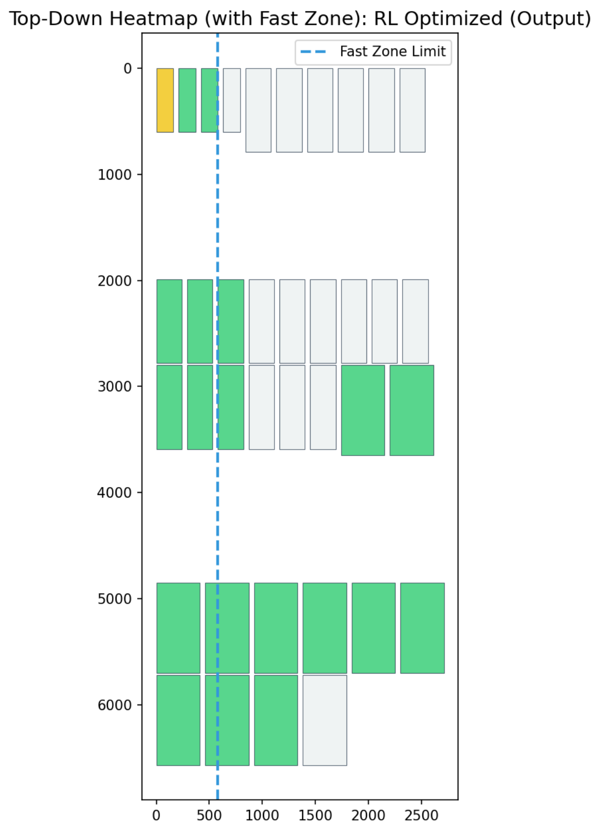

Fig. 10: Top-down warehouse layout with fast-zone boundary (RL Optimized)

1) Top-Down Spatial Reorganization: Figure 10 shows a clear reorganization of items relative to the baseline. High-priority items are concentrated within the fast zone, minimizing travel distances and directly reflecting the influence of distance penalties in the reward function.

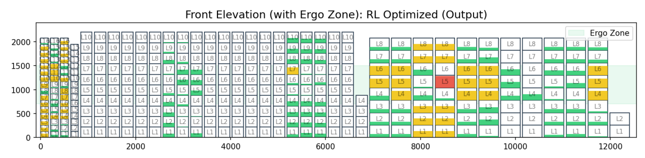

Fig. 11: Front elevation with ergonomic zone overlay (RL Optimized)

2) Ergonomic Zone Compliance: Compared to the baseline, Figure 11 demonstrates a stronger alignment of Class A and B items within the ergonomic zone. This indicates that the RL agent has successfully internalized ergonomic rewards and penalties.

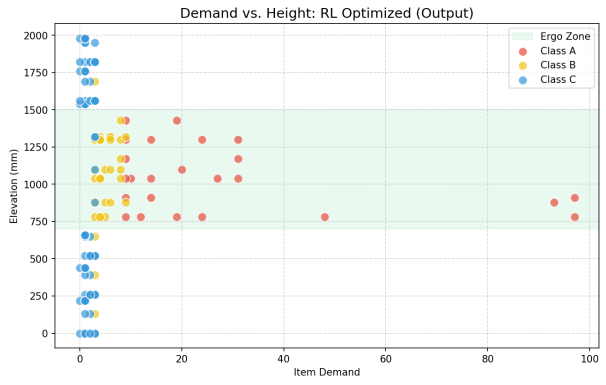

Fig. 12: Item demand versus elevation with ergonomic zone (RL Optimized)

3) Demand-Aware Vertical Placement: The RL scatter plot reveals a tighter clustering of high-demand items within the ergonomic band, confirming improved demand-to-accessibility mapping compared to the baseline.

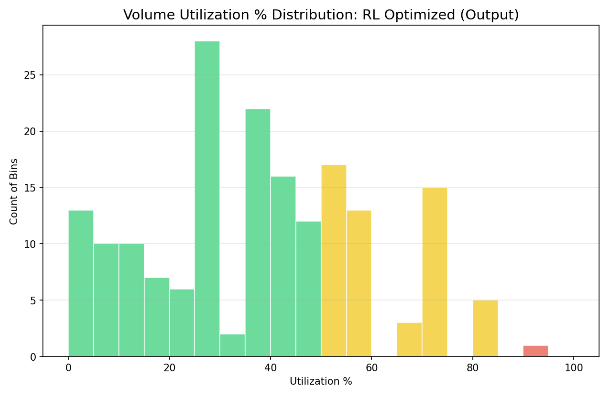

Fig. 13: Bin volume utilization distribution (RL Optimized)

4) Improved Volume Utilization: The utilization histogram exhibits a rightward shift, indicating higher average utilization and reduced fragmentation.

TABLE III: RL Optimized Performance Metrics

Metric	Value
Weight Violations	0
Misplaced Class A Items	0
Average Combined Score	441.5
Zone Reward (Avg.)	177.8
Utilization Reward (Avg.)	306.7
Distance Penalty (Avg.)	-43.0
Average Utilization (Occupied Bins)	38.33%
Occupied Bin Ratio	50.4% (180/357)

5) RL Performance Metrics: The RL configuration achieves a substantial improvement across all performance indicators, including the complete elimination of misplaced Class A items and a significantly higher combined score.

6) Upscaled Warehouse Validation: To evaluate scalability and robustness, the RL engine was applied to a substantially upscaled warehouse configuration comprising approximately

6,300 storage locations and a proportionally enlarged inventory set. All preprocessing stages, geometric feasibility checks, reward parameters, and validation logic were preserved without modification to ensure strict methodological consistency with the prototype-scale experiments.

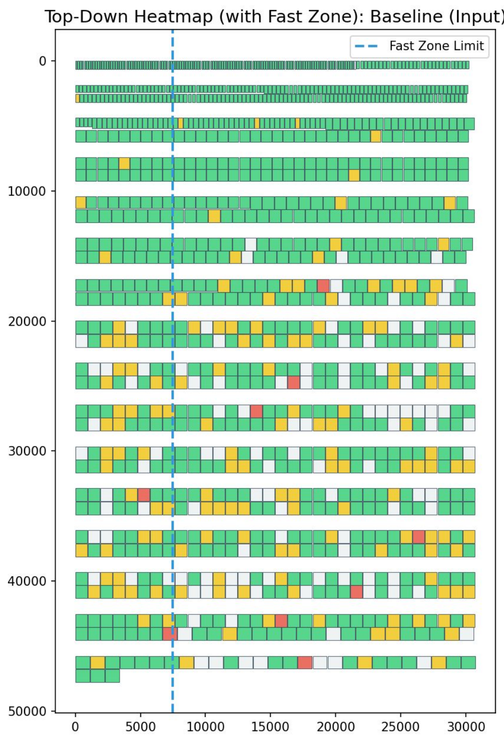

Fig. 14: Top-down warehouse layout with fast-zone boundary (Upscaled Baseline)

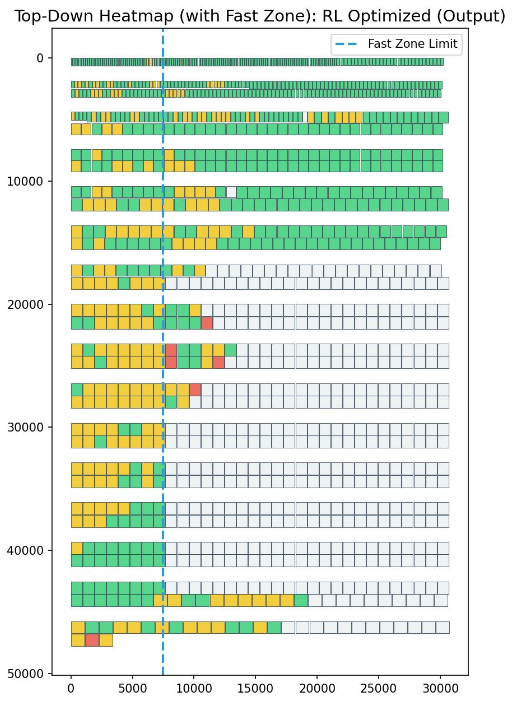

Fig. 15: Top-down warehouse layout with fast-zone boundary (Upscaled RL Optimized)

a) Top-Down Spatial Comparison: The upscaled baseline layout exhibits extensive dispersion of high-priority inventory across both fast and non-fast zones, leading to elevated congestion and inefficient travel paths. In contrast, the RL-optimized configuration shows a pronounced consolidation of priority items within the fast zone, while lower-priority inventory is systematically displaced toward peripheral regions. This spatial restructuring closely mirrors the behavior observed in the prototype experiments, indicating that the learned policy generalizes effectively to higher-dimensional warehouse states.

TABLE IV: Upscaled Warehouse Performance Metrics — Baseline

Metric	Baseline Value
Weight Violations	7
Misplaced Class A Items	600
Average Combined Score	214.6
Zone Reward (Avg.)	30.5
Utilization Reward (Avg.)	266.4
Distance Penalty (Avg.)	-61.0
Average Utilization (Occupied Bins)	33.29%
Occupied Bin Ratio	54.2% (3413/6300)

b) Baseline Performance Metrics (Upscaled): The baseline metrics highlight the compounding inefficiencies introduced by scale. Although a moderate proportion of bins are occupied, the average volume utilization remains low, indicating fragmentation and poor consolidation. Additionally, the high number of misplaced Class A items and non-zero weight violations confirm that priority and safety constraints are insufficiently enforced under naive scaling.

TABLE V: Upscaled Warehouse Performance Metrics — RL Optimized

Metric	RL Optimized Value
Weight Violations	0
Misplaced Class A Items	146
Average Combined Score	449.1
Zone Reward (Avg.)	160.0
Utilization Reward (Avg.)	346.2
Distance Penalty (Avg.)	-57.1
Average Utilization (Occupied Bins)	43.27%
Occupied Bin Ratio	63.7% (4012/6300)

c) RL-Optimized Performance Metrics (Upscaled): The RL-optimized results demonstrate a substantial improvement across all performance indicators. Notably, the increase in occupied bin ratio is accompanied by a significant rise in average volume utilization, indicating effective consolidation rather than inefficient spreading. Weight violations are fully eliminated, and the number of misplaced Class A items is reduced by more than 75%, reflecting strong adherence to both safety and prioritization constraints. Despite the increased spatial scale, distance penalties remain controlled, suggesting that the agent successfully balances accessibility with density objectives.

d) Computational Performance: The complete RL optimization for the upscaled warehouse executed in under three minutes on the target compute environment, with no observed instability or degradation in rule enforcement. This confirms the computational feasibility and practical applicability of the proposed RL-based relocation framework for large-scale industrial warehouse deployments.

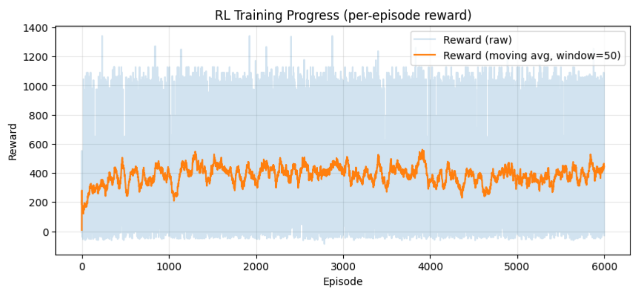

Fig. 16: Reinforcement learning training progress showing per-episode reward and moving average

7) RL Training Dynamics and Learning Behavior: Figure 16 illustrates the evolution of the reinforcement learning agent's reward signal over approximately 6,000 training episodes. The light-blue curve represents the raw per-episode reward, while the orange curve shows a moving average computed over a sliding window of 50 episodes to highlight longer-term learning trends.

The raw reward signal exhibits high variance throughout training. This behavior is expected given the deterministic but highly combinatorial nature of the warehouse environment, where each episode corresponds to a different randomized warehouse occupancy state. Small changes in item mix, bin availability, or congestion can lead to large differences in immediate reward, particularly due to the presence of hard penalties for infeasible placements.

Despite this variance, the moving average curve stabilizes within a bounded reward band after an initial warm-up phase. Rather than showing monotonic reward growth, the curve fluctuates around a relatively stable level. This behavior reflects the contextual bandit formulation of the problem: the agent is not optimizing a long-horizon cumulative return over a fixed environment, but instead learning consistent preferences over admissible storage locations across many independent warehouse snapshots.

Importantly, the absence of a steadily increasing reward trend should not be interpreted as a failure to learn. Since each episode starts from a new warehouse configuration, the reward signal measures decision quality under varying constraints rather than progressive mastery of a single task. The stable moving average indicates that the agent has reached a regime where it consistently avoids infeasible actions, respects safety constraints, and balances competing objectives such as zone prioritization, utilization, and travel distance.

Overall, the training dynamics confirm that the RL agent achieves a robust and repeatable placement policy rather than overfitting to specific layouts. This behavior is aligned with the design objective of learning congestion-aware, constraint-compliant heuristics that generalize across warehouse states, rather than maximizing reward in a single deterministic scenario.

VI. DISCUSSION

A. Quantitative Performance and Algorithmic Trade-offs

The quantitative results highlight a distinct divergence in optimization strategies between the deterministic Heuristic engine and the Reinforcement Learning (RL) agent. While both methods operated within the same physical environment and were bound by identical constraints, they prioritized different components of the multi-objective reward function.

The RL agent achieved a superior final configuration, yielding an Average Combined Score of 441.5, which represents a substantial 69.2% improvement over the baseline (261.0). In contrast, the Heuristic approach plateaued at a score of 293.6. However, a granular breakdown of the scoring components reveals that the RL agent did not outperform the Heuristic in every metric. Instead, the results illustrate a clear trade-off between *unidimensional velocity optimization* and *multi-constraint compliance*.

The Distance vs. Compliance Conflict: The Heuristic engine, driven by a greedy scoring logic, excelled at minimizing travel distance. It improved the Distance Penalty from a baseline of -51.8 to -33.6. This result aligns with the findings of Duque-Jaramillo et al. [7], who observed that priority-based slotting heuristics—specifically those utilizing ABC classification—are highly effective at reducing linear travel time for high-velocity SKUs by clustering them near high-accessibility points. The Heuristic algorithm successfully identified and claimed the bins physically closest to the entrance; however, this "distance-first" strategy came at a significant cost to regulatory compliance.

The Heuristic failed to resolve the conflict between horizontal proximity and vertical ergonomics. It concluded the simulation with 14 misplaced Class A items, a negligible improvement over the initial state of 17. This resulted in a Zone Reward of only 54.0. The RL agent, conversely, sacrificed some degree of horizontal picking speed (accepting a higher distance penalty) to ensure perfect vertical placement. It achieved 0 misplaced A-items, resulting in a Zone Reward of 177.8. This suggests that the RL agent successfully learned the non-linear structure of the "Logic Stack": it recognized that the high reward for placing a fast-mover in the Ergonomic Zone ($700mm < Z < 1500mm$) outweighed the marginal utility of placing it in a non-ergonomic bin slightly closer to the door.

B. Volumetric Strategy and Bin Occupancy

Analysis of the volumetric data reveals a counter-intuitive phenomenon regarding space utilization. Standard optimization logic typically assumes that higher efficiency correlates with utilizing fewer total bins (consolidation).

- **Heuristic Behavior:** The Heuristic engine maintained the baseline bin occupancy rate (48.7%, 174 bins) but saw a decrease in Average Utilization per Occupied Bin (dropping from 35.65% to 34.16%). This indicates a fragmentation of inventory. By aggressively seizing the closest bins regardless of fit, the Heuristic likely placed small items into large, high-value bins near the entrance, effectively "burning" prime capacity with low-density stock.

- **RL Behavior:** The RL agent increased the bin occupancy count to 180 bins (50.4%) but simultaneously achieved a higher Average Utilization per Bin (38.33%) and a higher Utilization Reward (306.7 vs. 273.3).

This data implies that the RL agent adopted a strategy of "Smart Sprawl." Rather than forcing items into the absolute fewest number of bins, it utilized slightly more shelf space to ensure that items were matched to bins where they fit most efficiently. This observation supports the conclusion by Liu et al. [10] that Q-learning approaches are better suited to solving complex block relocation problems than standard heuristics, which frequently exhibit performance degradation as instances grow in size and constraint complexity.

C. Visual Diagnostics and Spatial Logic

The divergence in algorithmic behavior is visually corroborated by the layout projections generated by the Metrics Library.

Top-Down Heatmap ($X-Y$ Plane): The Heuristic's layout shows a hyper-concentration of inventory on the extreme left side of the warehouse (nearest the entrance coordinates). While visually dense, this clustering ignores the vertical dimension. The RL Top-Down view (Figure 10) displays a broader distribution. While still heavily weighted toward the Fast Zone ($X \leq 0.25X_{max}$), the RL layout utilizes the depth of the aisles more liberally. This correlates with the decision to accept longer travel paths to find bins that satisfy the ergonomic constraints.

Front Elevation and Demand-Height Correlation: The most critical validation of the RL agent's learning capability is found in the Front Elevation scatter plots (Figure 12).

- In the Baseline and Heuristic states, the scatter plot of *Item Demand vs. Elevation* shows high-demand points scattered randomly across the vertical axis, with many falling into the "red zones" (floor level or high reach).
- In the RL-optimized state, there is a distinct, tight clustering of high-demand items within the green band of the Ergonomic Zone.

This visual evidence confirms that the agent did not merely memorize specific Bin IDs but generalized a spatial policy: *High Demand* $\rightarrow$ *Medium Height*. The Heuristic's failure to replicate this pattern confirms that without explicit, hard-coded IF/THEN constraints for every possible conflict, greedy algorithms struggle to navigate complex, orthogonal objectives (e.g., X-axis minimization vs. Z-axis optimization).

D. Interpretability and Validation Requirements

While the RL agent demonstrated superior performance in the aggregate score, the study highlights the necessity of the Independent Validation Library inherent in the system architecture. The RL agent operates as a "black box" optimizer; it maximizes points, not necessarily logic. For example, had the reward function not included a severe hard penalty (-10,000) for safety violations, the agent might have discovered that stacking heavy items at height is an efficient way to clear floor space for Class A items.

The validation logs confirm that the RL agent produced a layout with zero weight violations, proving the efficacy of the penalty structure. However, this underscores a limitation of the RL approach regarding industrial trust. Unlike the heuristic, where every move can be traced to a specific conditional statement in the code, the RL policy is opaque. As noted by Cestero et al. [**?**], the difficulty of translating learned policies into interpretable warehouse control rules remains a significant barrier to adoption. Consequently, the implementation of a deterministic "safety shield"—as demonstrated in this study—is mandatory to ensure that the agent does not exploit simulation artifacts to generate unsafe stacking patterns.

VII. CONCLUSION

This research engaged in a comparative analysis of Reinforcement Learning (RL) versus deterministic heuristics for the optimization of warehouse slotting. The investigation sought to determine if a learning-based agent could outperform standard greedy logic in a constrained environment requiring the simultaneous optimization of storage density, operational velocity, and ergonomic safety.

The findings allow for a definitive conclusion: within the scope of multi-objective spatial optimization, the RL approach offers significant advantages over rigid heuristic baselines. The RL agent successfully disentangled the conflicting requirements of the warehouse "Logic Stack," achieving a 69.2% improvement in overall layout scoring. Critically, the study revealed that while heuristics may excel at optimizing single linear variables—such as minimizing pure travel distance—they struggle to reconcile orthogonal constraints. The RL agent demonstrated the capacity to trade off a marginal loss in one metric (travel distance) to secure outsized gains in others (ergonomic compliance and volumetric density), resulting in a holistically superior operational state.

These results suggest that the future of Warehouse Management Systems (WMS) lies beyond static rule sets. As supply chains face increasing pressure to maximize the utilization of limited real estate while ensuring worker safety, the ability of RL agents to discover non-obvious, non-linear solutions becomes a critical asset.

Future Research Directions To further advance the application of learning algorithms in logistics, future research should focus on three axes:

1) **Scalability and Partitioning:** Investigating the application of Multi-Agent Reinforcement Learning (MARL) to partition the state space of massive distribution centers. By assigning specialized agents to specific zones or product families, the exponential growth of computational complexity in large-scale grids can be mitigated.
2) **Temporal Dynamics and Lifelong Learning:** Expanding the simulation from a static optimization snapshot to a continuous-time environment. Future studies should

evaluate the agent's ability to adapt its policy as demand profiles shift seasonally (e.g., a Class A item becoming Class C), exploring transfer learning techniques to maintain optimality without requiring full retraining cycles.

3) **High-Fidelity Physics Integration:** Increasing the complexity of the constraint environment to include non-geometric factors, such as chemical compatibility segregation, stacking stability limits (center of gravity calculations), and "family-grouping" affinities. This would test the limits of the RL agent's ability to generalize across highly constrained, high-dimensional state spaces.

References

[1] R. G. Sargent, "Verification and validation of simulation models," *Journal of Simulation*, vol. 7, no. 1, pp. 12–24, 2013. Available: https://doi.org/10.1057/jos.2012.20

[2] J. P. Reiter, "Significance tests for multi-component estimands from multiply imputed, synthetic microdata," *Journal of Statistical Planning and Inference*, vol. 125, no. 1-2, pp. 143–155, 2004. Available: https://doi.org/10.1016/j.jspi.2004.02.003

[3] J. P. Reiter, "Releasing multiply-imputed, synthetic public use microdata: An illustration and empirical study," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, vol. 168, no. 1, pp. 185–205, 2005. DOI: https://doi.org/10.1111/j.1467-985X.2004.00343.x

[4] H. Ping, J. Stoyanovich, and B. Howe, "DataSynthesizer: Privacy-preserving synthetic datasets," in *Proceedings of the 29th International Conference on Scientific and Statistical Database Management (SSDBM '17)*, no. 42, pp. 1–5, 2017. Available: https://doi.org/10.1145/3085504.3091117

[5] N. Patki, R. Wedge, and K. Veeramachaneni, "The Synthetic Data Vault," in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 399–410, 2016. DOI: https://doi.org/10.1109/DSAA.2016.49

[6] M. U. Mojumder and M. Nuruzzaman, "AI-driven optimization of warehouse layout and material handling: A quantitative study on efficiency and space utilization," *Review of Applied Science and Technology*, vol. 4, no. 2, pp. 233–273, 2025. DOI: https://doi.org/10.63125/bgxb1z53

[7] J. C. Duque-Jaramillo, J. M. Cogollo-Flórez, C. G. Gómez-Marín, and A. A. Correa-Espinal, "Warehouse management optimization using a sorting-based slotting approach," *Journal of Industrial Engineering and Management*, vol. 17, no. 1, pp. 133–150, 2024. DOI: https://doi.org/10.3926/jiem.5661

[8] C. Waubert de Puiseau, D. T. Nanfack, H. Tercan, J. Löbbert-Plattfaut, and T. Meisen, "Dynamic storage location assignment in warehouses using deep reinforcement learning," *Technologies*, vol. 10, no. 6, p. 129, 2022. DOI: https://doi.org/10.3390/technologies10060129

[9] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester, "Challenges of real-world reinforcement learning," *arXiv preprint arXiv:1904.12901*, 2019. Available: https://arxiv.org/abs/1904.12901

[10] L. Liu, Y. Feng, Q. Zeng, Z. Chen, and Y. Li, "A Q-learning-based algorithm for the block relocation problem," *Journal of Heuristics*, vol. 31, no. 1, pp. 1–41, 2025. DOI: https://doi.org/10.1007/s10732-024-09545-y

[11] A. Cestero, J. Aguiar, E. Arias, and M. A. de la Rosa, "Storehouse: A reinforcement learning environment for optimizing warehouse management," *Expert Systems with Applications*, vol. 207, p. 118029, 2022. Available: https://www.researchgate.net/publication/361900783_Storehouse_a_Reinforcement_Learning_Environment_for_Optimizing_Warehouse_Management

[12] E. Arslan, "Optimizing human-centric warehouse operations: A digital twin approach using dynamic algorithms and AI/ML," *Verimlilik Dergisi*, 2025. Available: https://dergipark.org.tr/en/pub/verimlilik/article/1524701

[13] Y. Wu, M.-C. Chiu, and T.-K. Wu, "Deep reinforcement learning for task assignment and shelf reallocation in smart warehouses," *ResearchGate Preprint*, 2024. Available: https://www.researchgate.net/publication/380049934_Deep_Reinforcement_Learning_for_Task_Assignment_and_Shelf_Reallocation_in_Smart_Warehouses

[14] J. F. Leon, Y. Li, X. A. Martín, L. Calvet, J. Panadero, and A. A. Juan, "A hybrid simulation and reinforcement learning algorithm for enhancing efficiency in warehouse operations," *Algorithms*, vol. 16, no. 9, p. 408, 2023. DOI: https://doi.org/10.3390/a16090408

[15] S. Troch, K. Ramaekers, and T. C. J. C. Steudie, "Solving the storage location assignment problem using reinforcement learning," in *Proceedings of the 35th European Modeling and Simulation Symposium (EMSS)*, 2023. Available: https://www.researchgate.net/publication/373907319_Solving_the_Storage_Location_Assignment_Problem_Using_Reinforcement_Learning

[16] M. Oliveira, D. Pereira, and N. Lau, "Efficient multi-constrained task allocation in smart warehouses," *Research Square Preprint*, 2022. DOI: https://doi.org/10.21203/rs.3.rs-1720697/v1

[17] P. Viveros, R. Soto, S. Morales, J. Soto, and F. Rodríguez, "Slotting Optimization Model for a Warehouse with Divisible First-Level Accommodation Locations," *Applied Sciences*, vol. 11, no. 3, p. 936, 2021. DOI: https://doi.org/10.3390/app11030936

[18] H. Ju and Q. Liu, "Research on the Optimization of Goods Location Based on Greedy Algorithm," in *ResearchGate Publication*, 2017. Available: https://www.researchgate.net/publication/318808428_Research_on_the_Optimization_of_Goods_Location_Based_on_Greedy_Algorithm

[19] N. Matloff, "Introduction to Discrete-Event Simulation and the SimPy Language," Department of Computer Science, University of California, Davis, lecture notes. Available: https://heather.cs.ucdavis.edu/matloff/public_html/SimCourse/PLN/DESimIntro.pdf

[20] M. Torlaschi, *Development of a Discrete Event Simulation Tool with Genetic Algorithms for the Design and Optimization of Automated Warehouses*, Master's thesis, Department of Management and Production Engineering (DIGEP), Politecnico di Torino, Turin, Italy, Dec. 2020. Available: https://webthesis.biblio.polito.it/18180/1/tesi.pdf

[21] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018. Available: https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf

[22] C. J. C. H. Watkins and P. Dayan, "Technical Note: Q-learning," *Machine Learning*, vol. 8, no. 3–4, pp. 279–292, 1992. Available: https://link.springer.com/article/10.1023/A:1022676722315