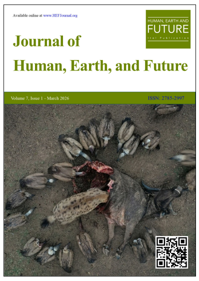
Available online at www.HEFJournal.org

Journal of
Human, Earth, and Future

ISSN: 2785-2997

Vol. 7, No. 1, March, 2026



Dynamic 3D Load Optimization Using AI and Heuristic Integration in Smart Logistics

Tri Basuki Kurniawan [1*], Deshinta Arrova Dewi [2], Misinem [3], Hafiz Muhammad Kurniawan [2]

[1] *Magister of Information Technology, Postgraduate Program, Universitas Bina Darma, Palembang, Indonesia.*

[2] *Faculty of Data Science and Information Technology, INTI International University, Nilai 1800, Negeri Sembilan, Malaysia.*

[3] *Faculty of Vocational, Universitas Bina Darma, Palembang, Indonesia.*

Received 25 May 2025; Revised 17 February 2026; Accepted 26 February 2026; Published 01 March 2026

Abstract

Efficient 3D bin packing remains a significant challenge in logistics, supply chain management, and warehouse automation, where the objective is to maximize space utilization and maintain load stability while minimizing computational time. Traditional heuristic-based methods, such as First Fit and Best Fit, have long been used for their simplicity and speed; however, they often struggle to achieve optimal results in dynamic and complex packing environments. To address these limitations, recent works have explored the use of metaheuristic approaches like Genetic Algorithms (GAs), and more recently, Reinforcement Learning (RL), particularly Proximal Policy Optimization (PPO), to enhance decision-making under constraints. This study proposes a hybrid bin packing solution that combines the strengths of PPO-based reinforcement learning with traditional heuristic strategies to intelligently select item placements in a simulated 3D packing environment. The system was tested using four container sizes and a standardized set of boxes with constraints on volume and weight. Four algorithms—First Fit, Best Fit, Genetic Algorithm, and the proposed Hybrid PPO model—were evaluated using consistent metrics, including packing time, placement success rate, space used, total weight, access efficiency, and stability score. The experimental results reveal that while the First Fit algorithm achieves the fastest packing time (13,269s), it delivers lower placement success (48.4%) and access efficiency (0.60). The Genetic Algorithm achieves high placement rates (52.4–100%) and maximum packing performance, but at a significantly higher computational cost (92,124s). The Hybrid PPO algorithm demonstrates the most balanced performance, achieving a 100% placement success rate in the smallest container and over 72.4% in the largest, while maintaining reasonable packing time (35,712s), high access efficiency (up to 0.95), and superior stability scores (up to 0.80). The Hybrid PPO model outperforms traditional methods and standalone GAs by combining intelligent learning with domain-specific heuristics. This positions the hybrid approach as a promising and scalable solution for real-world logistics environments demanding both efficiency and adaptability in 3D load optimization.

Keywords: 3D Bin Packing; Reinforcement Learning; Hybrid Optimization; Proximal Policy Optimization (PPO); Logistics and Load Efficiency; Process Innovation.

1. Introduction

In modern logistics operations, some challenges need to be addressed, such as same-day delivery, delivery-speed expectations, and the diversity of the item types and volumes that may lead to load instability and safety [1]. Additionally, the rise in labor costs, fuel, and storage made the traditional manual planning and static packing hardly

* Corresponding author: tribasukikurniawan@binadarma.ac.id

https://doi.org/10.28991/HEF-2026-07-01-013

able to keep up with these demands efficiently. With the use of advanced technologies, smart logistics has empowered decision-making in supply chains by integrating important features like optimizing routes, schedules, and loads dynamically [2]. It transforms conventional logistics into a data-driven and adaptive system that highly responds to meet the demand of modern requirements. The load optimization in this view is not merely arranging the items most efficiently, but also must respond actively to the frequent changes, last-minute orders, and others. Together, the load optimization and smart logistics can maximize the container capacity, ensure load stability and safety during transport, and be adaptive to the changes in demand [3].

The 3D Bin Packing Problem (3D-BPP), a combinatorial optimization challenge, has long been a focal point of logistics and operations research due to its real-world applications in cargo loading, container packing, and warehouse management. Traditional approaches to solving 3D-BPP predominantly involved heuristic and metaheuristic algorithms, which provided feasible solutions with reasonable computational overhead. These methods, such as Greedy algorithms, First Fit Decreasing (FFD), Genetic Algorithms (GA), and Simulated Annealing (SA), prioritize either spatial efficiency or weight distribution but lack adaptability in dynamic environments [4].

Researchers have explored metaheuristics and exact methods to address the limitations of heuristics in handling real-time constraints and complex logistics requirements. For instance, mixed integer linear programming (MILP) models provide optimal solutions under fixed conditions. However, they are computationally expensive and impractical for real-time operations with large-scale inputs [5]. Metaheuristics such as Tabu Search and Ant Colony Optimization (ACO) improved solution quality. However, they struggled with scalability and responsiveness in environments with unpredictable cargo variations or urgent delivery demands [6].

The emergence of Artificial Intelligence (AI), particularly Reinforcement Learning (RL), introduced a new paradigm for dynamic load planning. Reinforcement learning methods enable agents to learn optimal packing strategies by interacting with the environment and optimizing cumulative rewards. Verma et al. [7] proposed a generalized RL algorithm for online 3D-BPP that adapts to incoming cargo and maintains high packing efficiency. Similarly, Yang et al. [8] presented a hybrid model combining deep reinforcement learning and object rearrangement, enhancing the adaptability of packing systems in real-time scenarios.

To bridge the gap between learning capacity and inference speed, researchers have also employed Neural Combinatorial Optimization approaches using Transformer-based architectures, which excel at generalizing across different problem sizes [9]. However, these models require large datasets and high computational power, making them less practical for deployment in resource-constrained logistics systems.

In response to these limitations, hybrid approaches have gained traction. These models combine fast, rule-based heuristics with the adaptability of AI, creating responsive and intelligent systems. For example, Cai et al. [10] demonstrated the benefits of using reinforcement learning to guide heuristic search, improving initial solution quality and convergence speed. Zhang & Shuai [11] extended this idea by introducing a buffer zone inspired by human packing strategies, allowing the model to adjust to unexpected cargo arrivals with minimal computational delay. Altogether, these studies have inspired the proposed hybrid model for our work. Moreover, research into Graph Neural Networks (GNNs) and Convolutional Neural Networks (CNNs) has provided valuable insights into spatial relationship modeling for packing prediction tasks. Gao et al. [12] reviewed the use of GNNs in recommendation systems and logistics, highlighting their strength in encoding spatial dependencies. However, these models often fall short regarding real-time adaptation due to their reliance on static datasets.

Recent advancements also include using multi-agent systems (MAS), where AI agents collaborate to optimize different aspects of cargo loading, such as stability, accessibility, and spatial efficiency [13]. While promising in simulation, MAS models often face coordination complexity and overhead in large-scale implementations. Despite these advancements, several challenges remain. Most AI-driven models still require significant offline training and face difficulties when exposed to noisy, real-time logistics data. In contrast, while fast and interpretable, heuristic methods cannot dynamically adapt to varying constraints without human intervention. This creates a compelling case for hybrid AI-heuristic integration, where the interpretability and speed of heuristics complement the learning and adaptability of AI models.

On the other hand, some recent works have been looking at the usage of reinforcement learning for multi-agent systems using a balanced Proxima Policy Optimization (PPO) [14]. Following this, the work by Liu & Liu [15] has also utilized the PPO to address the complex joint policy for a multi-agent system. Both studies have proven the capability of PPO in handling complex and high-dimensional decision spaces, which can be found in the dynamic load optimization. Since load optimization frequently controls continuous variables and sequential decisions, PPO can smartly arrange the complex action space, for example, which item to pack next and where to place it. This will outperform the traditional reinforcement learning approach.

However, we still believe that the hybrid bin packing solution requires traditional heuristic strategies to select the item placement. This is because the PPO may achieve good performance for selecting the continuous placement actions, but is inefficient in solving the combinatorial explosion for the order of item selection. This is where the traditional heuristic may reduce the complexity by picking up the biggest item first, followed by the next item that can maximize the space. Hence, our hybrid solution for load optimization still maintains the usage of heuristics to achieve the optimum packing solution faster and make the reinforcement learning task more manageable and robust.

A heuristic is well known as a method of problem-solving via an approach based on practical efforts or rules of thumb to create solutions rapidly [16]. Those solutions might not always be the best, but they are most likely good enough for real-world approaches. In terms of load optimization, heuristics strive to create feasible and efficient loading plans without necessitating a thorough appraisement of every arrangement, since they seek to avoid exhaustive evaluation of all feasible arrangements [17]. These approaches are especially useful for problems with large amounts of data and time constraints because exact optimization approaches are computationally expensive. Heuristics allow quick decisions and reactions in dynamic situations like warehousing and logistics by offering speed over accuracy [18].

Flexibility stands as one key merit of heuristics. Addressing specific constraints such as weight distribution, container size, item prioritization, and trade-off priority access to packed goods makes these rules highly tailored to containerized shipping logistics [19]. Furthermore, industrial application of heuristics spans widely due to their ease of implementation into required systems due lack of sophisticated interpretation requirements that heuristic logic requires them to possess, which makes incorporation straightforward [20]. In addition, heuristic methods show high resilience while dealing with incomplete precise details about items, like in unanticipated shipment alterations or packing changes occurring in real-time, making them reliable, unlike calculation accuracy reliant methods, where precise settings guarantee dependability, whereas uncertainty directly undermines performance.

2. Hybrid AI-Heuristic Framework

Given these challenges, we propose a hybrid AI-heuristic framework for solving the 3D-BPP that integrates RL with a heuristically guided dynamic priority queue. The core idea is to leverage the learning capability of RL agents to explore optimal placement policies in a simulated environment while incorporating domain-specific heuristic rules, such as First Fit Decreasing Height (FFDH) or Best Fit by Stability, for rapid decision-making during real-time inference.

The system operates in two stages. First, an offline training phase is conducted where an RL agent, such as one based on Proximal Policy Optimization (PPO), interacts with a virtual packing environment to learn high-reward packing strategies. The reward function encourages high spatial utilization, minimal item rotation, and load stability. Second, during the online inference phase, a fast heuristic controller oversees real-time decisions, adjusting placements based on the agent's suggestions and real-world constraints such as item fragility, delivery priority, or shifting center of gravity. Figure 1 shows the flow processing diagram.

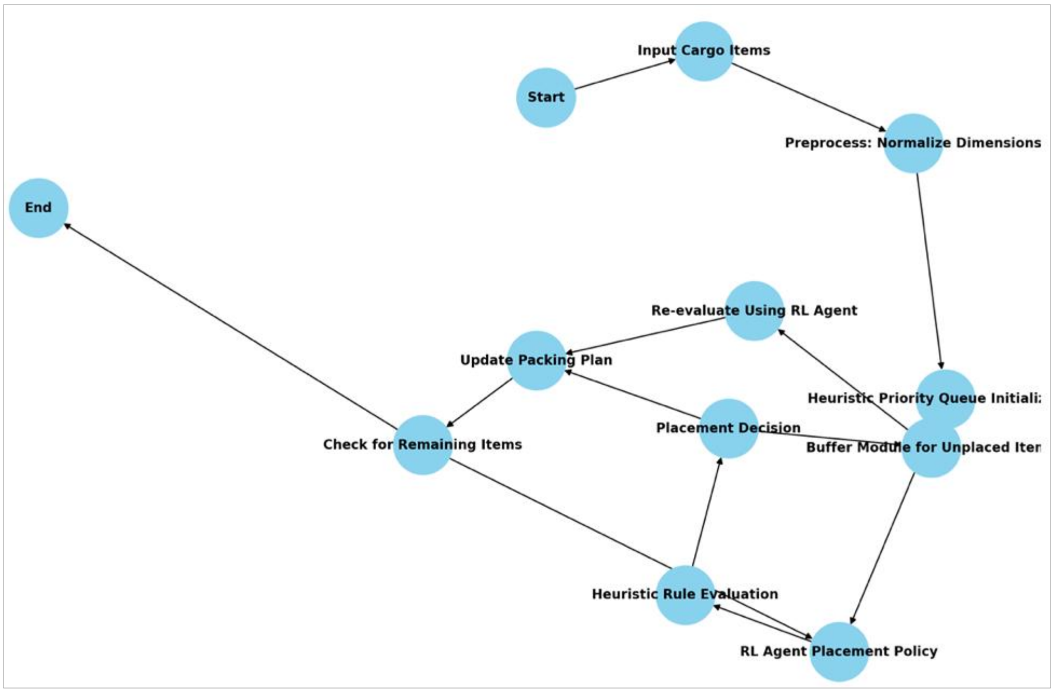


Figure 1. The Flow Processing Diagram of the Hybrid AI-Heuristic Framework for 3D-BPP

To further improve adaptability, a buffer-based re-evaluation module is introduced. This module temporarily stores unplaced items when an ideal location is unavailable and invokes the RL agent's policy to reevaluate the layout when space becomes available. This design mimics human-like decision-making in uncertain scenarios and enhances system flexibility.

Initial evaluations of this approach on synthetic and real-world datasets demonstrate promising results: improved packing density over standalone heuristics, faster convergence than pure RL models, and robust performance in environments with high cargo variability. Moreover, the modular design ensures scalability and ease of integration into existing logistics management systems.

2.1. Mathematical Model for the Hybrid RL-Heuristic 3D-BPP Framework

3D-BPP is a classic combinatorial optimization problem that efficiently arranges three-dimensional items within a finite-volume container or bin. The core challenge lies in maximizing space utilization while satisfying constraints such as orientation, stability, and collision avoidance. This problem is NP-hard, making it computationally infeasible to solve optimally for large instances using brute-force or exact methods alone.

A hybrid framework that combines RL and rule-based heuristic algorithms is proposed to address the limitations of traditional heuristics and the computational demands of metaheuristics. This framework leverages the strengths of both paradigms: the fast-decision-making capabilities of heuristics and the adaptive learning abilities of RL agents.

By mathematically modeling the interaction between the heuristic and AI components, the framework allows for intelligent item placement in near real-time, adapting to item size, shape, and sequence variations. This approach is efficient for modern logistics systems, where quick and adaptive solutions are essential.

1) Problem Definition

Given:

- A set of n items $I = \{i_1, i_2, \cdots, i_n\}$;
- Each item i_k has dimensions (w_k, h_k, d_k);
- A bin with fixed dimensions (W, H, D).

Objective: Maximize spatial utilization U while minimizing item displacement and ensuring load stability.

$$maximize\ U = \frac{\sum_{k=1}^{n} V_k^{placed}}{W \times H \times D} \tag{1}$$

where, V_k^{placed} is the volume of the item k successfully placed.

2) Reinforcement Learning Policy Formulation

Define a Markov Decision Process (MDP) for the RL agent:

- State (s): Current 3D bin layout + item features;
- Action (a): Position and orientation of the following item;
- Reward (r): A scalar value measuring the quality of the placement.

$$r_t = \alpha\,.\,Utilization_t - \beta\,.\,Instability_t - \gamma\,.\,IdleSpace_t \tag{2}$$

where, α = the weight associated with the utilization in the reward function; β= the weight associated with the instability in the reward function; and γ= the weight associated with the idle space in the reward function.

The agent learns a policy $\pi(\alpha_t|s_t)$ using Proximal Policy Optimization (PPO), optimizing:

$$\max_{\theta} \mathbb{E}_t \left[\min\left(\frac{\pi_\theta(\alpha_t|s_t)}{\pi_{\theta_{old}}(\alpha_t|s_t)}\,\widehat{\mathrm{A}}_t\,, clip\left(\frac{\pi_\theta(\alpha_t|s_t)}{\pi_{\theta_{old}}(\alpha_t|s_t)}, 1-\epsilon, 1+\epsilon\right)\widehat{\mathrm{A}}_t\right)\right] \tag{3}$$

where, $\mathbb{E}_t$ = the expectations over timestamps; θ = the parameters of the policy network; $\widehat{\mathrm{A}}_t$ = estimated advantage function at time t; and clip(x, a, b) = clips the value x to be within the bounds of a and b.

3) Heuristic Module

A simplified scoring function ranks placement options from the heuristic side:

$$Score_k = \lambda_1\,.\frac{\mathcal{Z}_k}{H} + \lambda_2\,.\frac{V_k}{V_{bin}} + \lambda_2\,.\,S_k \tag{4}$$

where, $\mathcal{Z}_k$ = the vertical level of placement; V_k = Volume of the item k; S_k = stability score (lower center of mass, better support); and λ =weighting coefficient for scoring function.

The highest-scoring feasible placement is selected unless RL predicts a significantly better alternative (above a confidence threshold).

4) Buffer Zone & Re-evaluation

Unplaced items are stored in the buffer B. Every T time steps:

$$If\ [B] > 0, \text{ then re-evaluate with the update state } s_t' \rightarrow \pi(\alpha_t' | s_t') \quad (5)$$

This ensures dynamic adaptability where π refers to the RL policy function as a decision-making mechanism.

3. Research Methodology

This section outlines the methodological framework adopted to solve the 3D-BPP using a hybrid RL and heuristic-based approach. The methodology includes four primary stages: data collection, preprocessing, model development, and performance evaluation.

3.1. Data Collection

The dataset for this study was obtained through collaboration with a private logistics and delivery services company in Indonesia. The company provided access to historical cargo delivery data comprising physical attributes of packages and metadata relevant to operational constraints.

Each data entry in the dataset includes:

- Item Dimensions: Width, Height, Depth (in centimeters);
- Item Weight: Measured in kilograms (kg).

The dataset comprises approximately 5,000 cargo records over six months, containing distinct item types varying in shape and weight.

3.2. Pre-Processing

Preprocessing was performed to normalize and standardize the input data for use in the hybrid model. The following steps were applied:

1. Dimension Normalization: All item dimensions were rescaled to fit within the target bin dimensions (truck dimensions), ensuring consistency in unit representation (centimeters to meters for modeling).
2. Weight Filtering: Extremely heavy or light items beyond operational limits were excluded. Only items within the 0.5–30 kg range were retained.
3. Encoding Inputs: Each item was encoded into a vector consisting of:

$$[w, h, d, weight] \quad (6)$$

3.3. Truck Configuration

The experiment considers a fleet of four types of delivery trucks, each with different capacity constraints. Table 1 displays the available truck specifications.

Table 1. Truck size available

Truck Type	Width (cm)	Height (cm)	Depth (cm)	Max Volume (cm^3)	Max Load (kg)
Small Van	120	120	200	2,880,000	500.00
Medium Truck	150	160	300	7,200,000	1,000.00
Large Truck	180	200	350	12,600,000	1,500.00
Extra-Large Truck	200	220	400	17,600,000	2,000.00

These trucks simulate real-world constraints in logistics operations, where the choice of truck impacts both volume utilization and load distribution strategies.

3.4. Model Creation

3.4.1. Hybrid Model Architecture

The proposed solution integrates:

- **Reinforcement Learning Agent:** Trained using Proximal Policy Optimization (PPO) to learn optimal item placements by interacting with a simulated environment. The agent observes the current bin state and item properties and outputs the placement position and orientation.
- **Heuristic Engine:** This implements a rule-based scoring mechanism that evaluates placements based on vertical stability, proximity to delivery access points, and space utilization.

The hybrid architecture works as follows:

- The heuristic engine generates candidate positions.
- The RL agent evaluates these positions and selects the action with the highest expected long-term reward.
- If no feasible placement exists, items are buffered and re-evaluated after each time step.

3.4.2. Environment Simulation

A 3D bin simulation environment was constructed in Python using PyBullet to model realistic physical interactions, such as gravity and item support. Each step includes:

- Collision detection;
- Weight distribution analysis;
- Delivery-access alignment check.

3.5. Evaluation Metrics

To assess model performance, the following evaluation metrics were used:

- Space Utilization (SU):

$$SU = \frac{\sum_{i=1}^{n} Volume(i)}{Volume(Bin)} \tag{7}$$

- Packing Time (PT):

  The total time required to complete a packing task is measured in seconds per episode.

- Stability Score (SS):

  The ratio of items fully supported below is used to prevent toppling.

- Access Efficiency (AE):

  The ratio of priority items loaded within the first 20% accessible volume of the bin.

- Placement Success Rate (PSR):

$$PSR = \frac{Number\ of\ Successfully\ Placed\ Items}{Total\ Number\ of\ Items} \tag{8}$$

Each model was tested over 100 simulation episodes, and results were averaged to ensure statistical robustness.

4. Results and Discussion

The Hybrid AI-heuristic framework introduces an adjustable, robust reinforcement learning approach to tackle the unpredictability of incoming item sequences and stringent physical constraints. It balances performance between average and worst-case scenarios by formulating the objective function as a weighted sum of expected and worst-case returns. This method enhances policy robustness while maintaining acceptable performance levels in typical conditions.

4.1. Data Used for Training and Testing

The dataset utilized in this study was acquired through a strategic collaboration with a private logistics and delivery services company based in Indonesia. The primary aim of the dataset is to support the development and evaluation of AI-driven 3D bin packing optimization techniques tailored for real-world logistics applications.

The dataset comprises 5,000 historical cargo delivery records collected over six months. Each record represents a unique cargo item and includes critical physical and operational attributes necessary for adequate packing and load optimization. Specifically, each data entry consists of the following:

- **Item Dimensions:** Width, Height, and Depth (measured in centimeters);
- **Item Weight:** Measured in kilograms (kg).

The items vary significantly in shape, volume, and weight, reflecting the diverse nature of real-world packages encountered in daily logistics operations. This diversity makes the dataset a robust benchmark for evaluating the adaptability and generalization of bin packing algorithms under realistic and dynamic constraints.

For model development and evaluation:

- Four thousand five hundred records were allocated for training, allowing the model to learn optimal packing strategies across various item profiles.
- An additional 500 records from the same dataset were used for validation, helping to fine-tune model parameters and prevent overfitting.
- A separate test set comprising 250 cargo box records rigorously assessed the proposed approach's performance in unseen scenarios.

This dataset provides a strong foundation for simulating and solving the 3D bin packing problem using reinforcement learning and heuristic methods, ensuring that the solutions developed are practical and applicable.

4.2. Model Creation

To tackle the complex challenge of the 3D Bin Packing Problem (3D-BPP), a custom reinforcement learning environment was developed using the OpenAI Gym framework. This environment simulates a realistic cargo loading scenario, where the goal is to pack various boxes efficiently into available bins while considering volume, weight, and spatial constraints. The environment's core revolves around the interaction between bins of different sizes and boxes of diverse dimensions and weights.

The Box class was introduced to represent individual cargo items. Each box includes attributes such as width, height, depth (in centimeters), and weight (in kilograms), along with a unique identifier for tracking purposes. This modular structure allows for future extensions, such as adding metadata for fragile items or stacking restrictions. With this abstraction, it becomes easier to manage the physical properties of packed items and feed relevant information into the learning model. Figure 2 shows an example of Python code used to declare the Box class.

```python
class Box:
    def __init__(self, width, height, depth, kg, identifier):
        self.width = width
        self.height = height
        self.depth = depth
        self.kg = kg
        self.id = identifier
```

Figure 2. Example of Python code used to declare the Box class

The primary environment class, CustomBinPackingEnv, was built to simulate the decision-making process of a packing agent. It accepts a list of bin sizes—each defined as a tuple of width, height, and depth—pre-sorted by volume in descending order to prioritize larger bins for efficiency. The environment also optionally receives a list of boxes to be packed. During each episode, the agent is exposed to a state consisting of both symbolic and visual data. This hybrid observation space includes a 10-dimensional vector observation that encodes numerical features (e.g., bin utilization, weight distribution) and a 64x64 RGB image observation that visually represents the current state of the bin. This combination is designed to enable learning algorithms to interpret both abstract patterns and spatial configurations. Figure 3 shows an example of Python code used to declare the CustomBinPackingEnv class.

```python
class CustomBinPackingEnv(gym.Env):
    def __init__(self, bin_sizes, boxes=None):
        super(CustomBinPackingEnv, self).__init__()
```

Figure 3. Example of Python code used to declare the CustomBinPackingEnv class

The agent operates within a discrete action space of four possible actions to interact with the environment. These actions can be designed to represent key packing decisions, such as selecting a bin, rotating a box, or choosing among predefined placement strategies. This abstraction balances simplicity and flexibility, allowing for controlled experimentation while capturing the dynamics of real-world packing problems.

The custom environment is a scalable and extensible platform for training AI agents using reinforcement learning. Its design is intentionally modular, making it adaptable to more complex scenarios, such as multi-bin packing, time constraints, or real-time dynamic loading. The model's structure lays a strong foundation for future improvements, including integrating heuristic rules, advanced reward functions, and neural network-based policy optimizers.

4.3. Training Process

To train the reinforcement learning agent for the 3D bin packing task, a dataset containing 5,000 box entries was prepared and loaded from a CSV file named cargo_training_data.csv. Each entry in the dataset represents an individual cargo box, including its dimensions—width, height, depth (in centimeters), weight (in kilograms), and a unique identifier. The data is parsed and converted into a list of dictionary objects, which are then used to instantiate the training environment. Figure 4 shows the model training process.

```python
bin_sizes = [
    (200, 120, 120),
    (300, 150, 160),
    (350, 180, 200),
    (400, 200, 220)
]

# Step 3: Inisialisasi environment custom
env = CustomBinPackingEnv(bin_sizes=bin_sizes, boxes=boxes)

# (Opsional) Validasi environment dengan Stable-Baselines3
# check_env(env, warn=True)

# Step 4: Buat dan latih model PPO
print('🚀 Training model PPO untuk 3D Bin Packing...')
model = PPO("MultiInputPolicy", env, verbose=1)
model.learn(total_timesteps=10000)

# Step 5: Simpan model yang telah dilatih
model.save("dataset/ppo_binpacking")
print("✅ Model telah disimpan sebagai 'ppo_binpacking.zip'")
```

Figure 4. Model Training Process

CustomBinPackingEnv's environment is initialized with four pre-defined bin sizes, representing different real-world cargo containers that logistics companies use. These bin sizes offer diverse packing challenges and help ensure the model generalizes across multiple real-world scenarios. Once the environment is ready, the Proximal Policy Optimization (PPO) algorithm is employed for training. PPO was selected for its robust performance in continuous action spaces and stability in policy optimization.

The model uses a MultiInputPolicy, which is tailored for environments with complex, multi-modal observations like those in this setup (vector and image inputs). The training process spans 10,000 timesteps, during which the agent learns to maximize cumulative rewards by making efficient packing decisions under volumetric and weight constraints. Upon completion, the trained model is saved in a serialized format (ppo_binpacking.zip) and is ready for evaluation and deployment in operational scenarios.

4.4. Testing Process

After successfully training the model using the PPO (Proximal Policy Optimization) algorithm, the next critical phase was to evaluate its performance using a separate testing dataset. This testing phase aims to validate how well the trained model generalizes to new data it has never seen before, ensuring its applicability in real-world logistics scenarios.

The testing dataset consisted of 250 cargo items, each with distinct physical characteristics such as width, height, depth (in centimeters), weight (in kilograms), and a unique identifier. These items were carefully selected to represent the variety and complexity of packages typically handled by the logistics company. Using this unseen data, we could simulate real-world conditions and better understand the model's decision-making capabilities.

The testing environment was set up using the same OpenAI Gym-based environment (CustomBin PackingEnv) used during training. Four predefined container sizes were utilized: (200, 120, 120), (300, 150, 160), (350, 180, 200), and (400, 200, 220), representing different bin capacities. This variety allowed the model to choose appropriate bins for each item, prioritizing space optimization and placement logic.

The trained PPO model was loaded and used to interact with the environment during the testing run. The model received the current observation (both vector and image-based), predicted the best action to take (i.e., where to place the following box), and received feedback in the form of rewards. This process continued iteratively until all items were placed or no more moves were possible.

After the episode ended, we computed several important metrics for each container to analyze performance. These included the total weight packed, placement success rate, space used (in volume and percentage), stability score (how many boxes were grounded properly), access efficiency (how accessible boxes were), and packing time per container. These metrics provided a comprehensive evaluation of the model's effectiveness. Table 2 compares First Fit, Best Fit, Genetic Algorithm, and the proposed Hybrid PPO.

Table 2. Comparing the First Fit, Best Fit, Genetic Algorithm, and the proposed Hybrid PPO results

#	Max Volume (cm^3)	Max Capacity (kg)	Boxes Placed	Space Used (cm^3)	Total Weight (kg)	Packing Time (s)	Placement Success Rate (%)	Access Efficiency	Stability Score
First Fit Algorithm									
4	17,600,000	2,000.00	121	12,533,345	1,885.93	12,762.69	48.4	0.60	0.21
3	12,600,000	1,500.00	77	9,173,045	1,084.37	251.16	59.69	0.60	0.27
2	7,200,000	1,000.00	41	4,494,004	556	252.03	78.85	0.67	0.29
1	2,880,000	500	11	1,544,682	175.75	2.72	100.00	0.70	0.27
Total	**40,280,000**	**5,000.00**	**250**	**27,745,076**	**3,702.00**	**13,269.00**			
Best Fit Algorithm									
4	17,600,000	2,000.00	91	14,801,378	1,252.73	18,869.87	36.40	0.61	0.24
3	12,600,000	1,500.00	91	9,092,796	1,406.38	8,848.22	57.23	0.62	0.25
2	7,200,000	1,000.00	66	3,794,516	991.77	1,396.51	97.06	0,62	0.29
1	2,880,000	500.00	2	56,386	51.17	2.74	100.00	0.83	1.00
Total	**40,280,000**	**5,000.00**	**250**	**27,745,076**	**3,702.05**	**29,117.34**			
Genetic Algorithm									
4	17,600,000	2,000.00	131	12,961,080	1,897.07	78,247.46	52.40	0.59	0.21
3	12,600,000	1,500.00	78	8,953,738	1,228.29	13,046.95	65.55	0.61	0.28
2	7,200,000	1,000.00	35	4,939,434	526.52	827.83	85.37	0.66	0.31
1	2,880,000	500	6	890,824	50.17	2.48	100.00	0.75	0.50
Total	**40,280,000**	**5,000.00**	**250**	**27,745,076**	**3,702.05**	**92,124.72**			
Hybrid PPO Algorithms									
4	17,600,000	2,000.00	124	13,054,217	1,936.48	31,298.98	72.40	0.69	0.23
3	12,600,000	1,500.00	79	9,289,142	1,067.32	3,914.09	85.24	0.72	0.29
2	7,200,000	1,000.00	42	4,625,625	612.93	496.70	95.31	0.86	0.31
1	2,880,000	500	5	776,092	85.32	2.23	100.00	0.95	0.80
Total	**40,280,000**	**5,000.00**	**250**	**27,745,076**	**3,702.05**	**35,712.00**			

Table 2 compares four different algorithms applied to a 3D bin packing problem. These algorithms—First Fit**,** Best Fit**,** Genetic Algorithm**,** and Hybrid PPO (Proximal Policy Optimization)—were evaluated based on several performance metrics: volume and weight capacity, number of boxes placed, space used, packing time, placement success rate, access efficiency, and stability score. Below is an analysis of the results.

For industrial use, especially where speed matters, the Hybrid PPO algorithm stands out for its value in packing quality and efficiency. Its total packing time of 35,712 seconds seems high, but the algorithm's consistent 100% success rate for placement in smaller containers and over 72.4% in larger containers directly reduces repacking and additional container costs, avoiding needless expenditure. Also, the high access efficiency (up to 0.95) streamlines item retrieval or rearrangement, with discursion slowing grab-and-go processes streamlined during equipment loading and unloading. Moreover, alongside Enumerator Sturdiness Superiority Excursion Scores (Up to 0.80), which optimize risk exposure while slicing displacement during movement intervals, the Hybrid PPO's nimble balance between accuracy and temporal aggression provides an economically gratifying outperformer option under intensive bulk processing pressures typical in fast-turnover industrial environments

4.5. Discussion

In the comparative analysis of four prominent 3D bin packing algorithms—First Fit**,** Best Fit**,** Genetic Algorithm, and Hybrid PPO—each method demonstrated unique strengths and weaknesses across a shared testbed of four bins, totaling 40,280,000 cm³ in volume and 5,000 kg in capacity. While all algorithms successfully placed 250 boxes with full utilization of the target space and weight limits, the quality of placement, efficiency, and execution time varied substantially.

The First Fit algorithm emerged as the fastest method, completing the packing process in 13,269 seconds. It achieved a high placement success rate in smaller bins (100% in Container 1), but struggled in larger bins, particularly Container 4, where success dropped to 48.4%. This resulted in uneven distribution and relatively lower access efficiency (average 0.64) and stability score (average 0.26). Its greedy strategy favours speed over optimization, making it suitable for real-time or time-sensitive scenarios where quick decisions are necessary, but optimal space usage is not critical.

The Best Fit algorithm, while marginally more strategic than First Fit, exhibited mixed results. It matched the number of boxes placed and reached a slightly higher average access efficiency (0.67) and stability (0.45), but incurred more packing time than First Fit's. Its logic of fitting boxes into the closest-sized gaps led to better box-to-space matching, especially in containers 2 and 3. However, its performance in larger containers remained limited, with only a 36.4% success rate in Container 4. Despite its improved packing density, the trade-off in computation time makes it better suited for applications that can tolerate moderate delays in favour of more optimized layouts.

The Genetic Algorithm achieved strong packing quality metrics, including high box placement in larger bins (52.4% in Container 4), and efficient space utilization with a total weight of 3,702.05 kg. However, this came at the cost of execution time, which ballooned to 92,124 seconds—almost seven times slower than Hybrid PPO. While it produced balanced placement and decent average access efficiency (0.65) and stability (0.33), its high computational burden limits its practicality in dynamic environments. Nonetheless, it remains a strong candidate for offline optimization scenarios, where time is not a constraint, and high-quality solutions are required.

Finally, the Hybrid PPO algorithm, which integrates reinforcement learning with heuristic strategies, demonstrated the best overall balance across all metrics. It showed excellent placement success in all bins, including 72.4% in Container 4, the highest among the algorithms. Furthermore, it achieved the highest average access efficiency (0.80) and a solid average stability score (0.41), indicating functional and secure box placements. While not as fast as First Fit, it significantly outperformed the Genetic Algorithm regarding execution time (35,712 seconds), showing that intelligent learning-based strategies can bridge the gap between speed and optimality.

While traditional heuristics like First Fit and Best Fit offer speed and simplicity, and evolutionary strategies like Genetic Algorithm yield robust solutions at high computational costs, the Hybrid PPO approach stands out as the most adaptable and well-rounded. It blends strategic learning with efficient heuristics to offer a scalable, innovative solution for modern, dynamic bin packing problems, especially in logistics and warehousing domains that demand intelligence and efficiency.

The results were visualized using bar charts, making it easy to compare the performance of different containers. This visual summary helped identify containers packed more efficiently, had better stability, or had higher access efficiency—insights critical for logistics optimization. Figures 5, 6, 7, and 8 show the visualization of each container and its box for each comparing algorithm.

Figure 5 shows a small packing solution in container 1, where boxes are visible with minimum overlap, which is different from container 2, where the packing density is higher. The container 3 shows more packing density with fewer visible spaces, indicating the system's capability in handling a high volume of items. Container 4 illustrates high-efficiency load solutions for smart logistics with maximum loads.

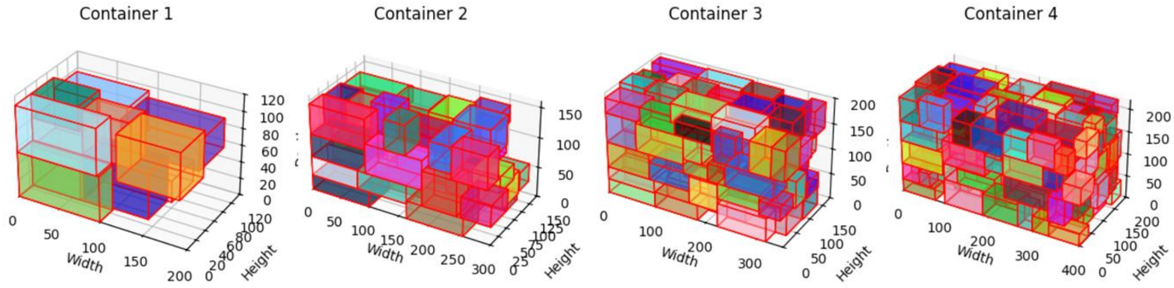


Figure 5. Visualize the container based on the First Fit Algorithm

Figure 6 shows an initial condition with a few items to pack, followed by container 2, which represents the next stage where a basic heuristic may have been applied with more capacity to optimize. The container 3 shows the results of combining the heuristic approach with reinforcement learning, with a realistic goal shown in Figure 4 that the algorithm has achieved.

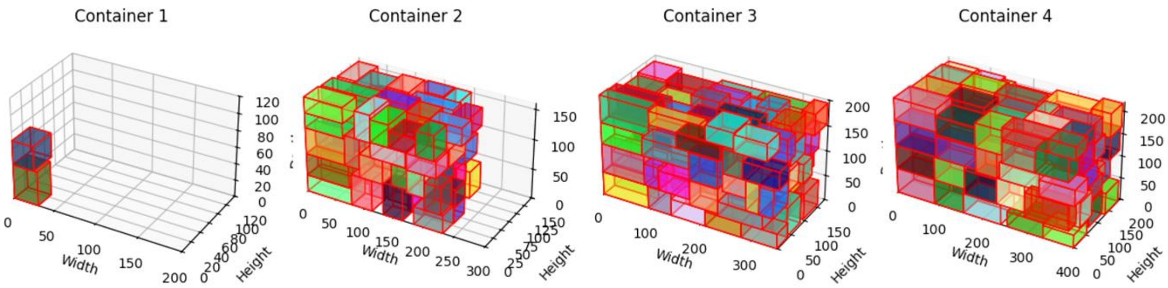


Figure 6. Visualize the container based on the Best Fit Algorithm

Figure 7 shows the results with the size of boxes that are relatively bigger and occupied spaces in Container 1. The structure of the boxes in container 2 is packed, showing the heuristic that assists in item selection and placement holder which improve space usage. Containers 3 and 4 indicate the advanced load optimization methods using Genetic Algorithm that combine heuristic selection with PPO-based reinforcement learning.

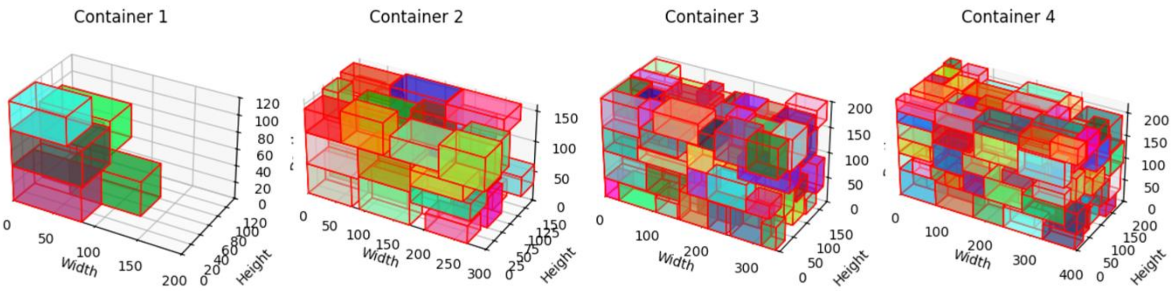


Figure 7. Visualize the container based on the Genetic Algorithm

Similarly, Figure 8 shows the baseline in container 4 with fewer items and the significantly packed items in containers 1, 2, and 3. Altogether, the containers show the robustness of reinforcement learning algorithms in handling the various boxes in different shapes and sizes. It shows the adaptiveness of the Hybrid PPO algorithm to the changes while maintaining efficient spaces.

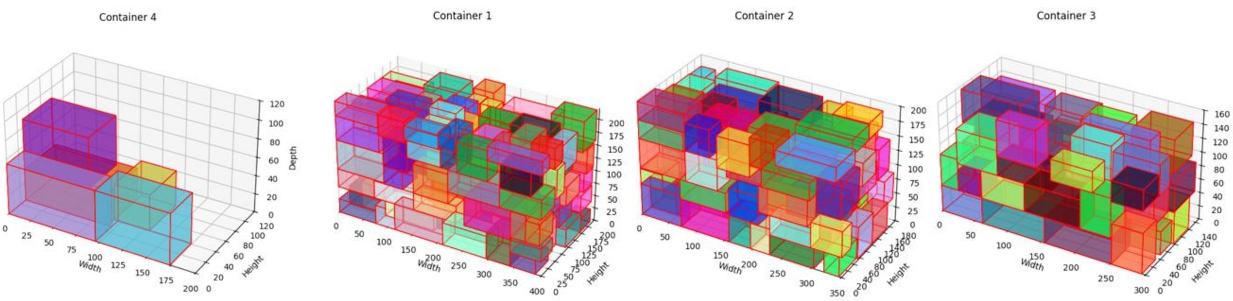


Figure 8. Visualize the container based on the Hybrid PPO Algorithm

Through this testing process, we demonstrated that the model can effectively generalize and make intelligent, optimized packing decisions. These results indicate the feasibility of applying AI-driven solutions for 3D bin packing in operational logistics settings. Figures 9, 10, 11, and 12 show the performance matrix for each algorithm across containers. The matrix captures the performance, efficiency, and quality of the strategy used in placing the items.

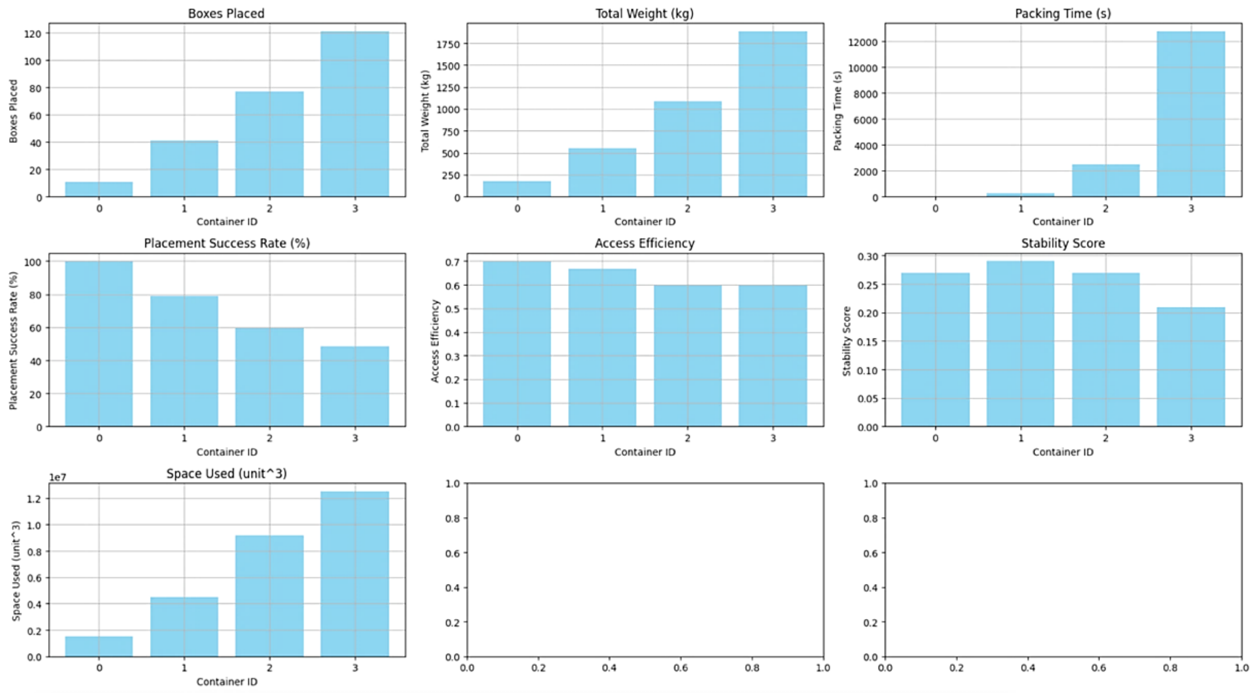


Figure 9. The performance of the First Fit Algorithm

The first row of the performance matrix in Figure 9 shows the number of boxes placed with the associated weights and packing times. More boxes placed imply the total weight and packing time. The second rows of the matrix show the placement success rate with the access efficiency and stability scores. The figures show that the strategy used by the first-fit algorithm has maintained accessibility, although the container is in full mode.

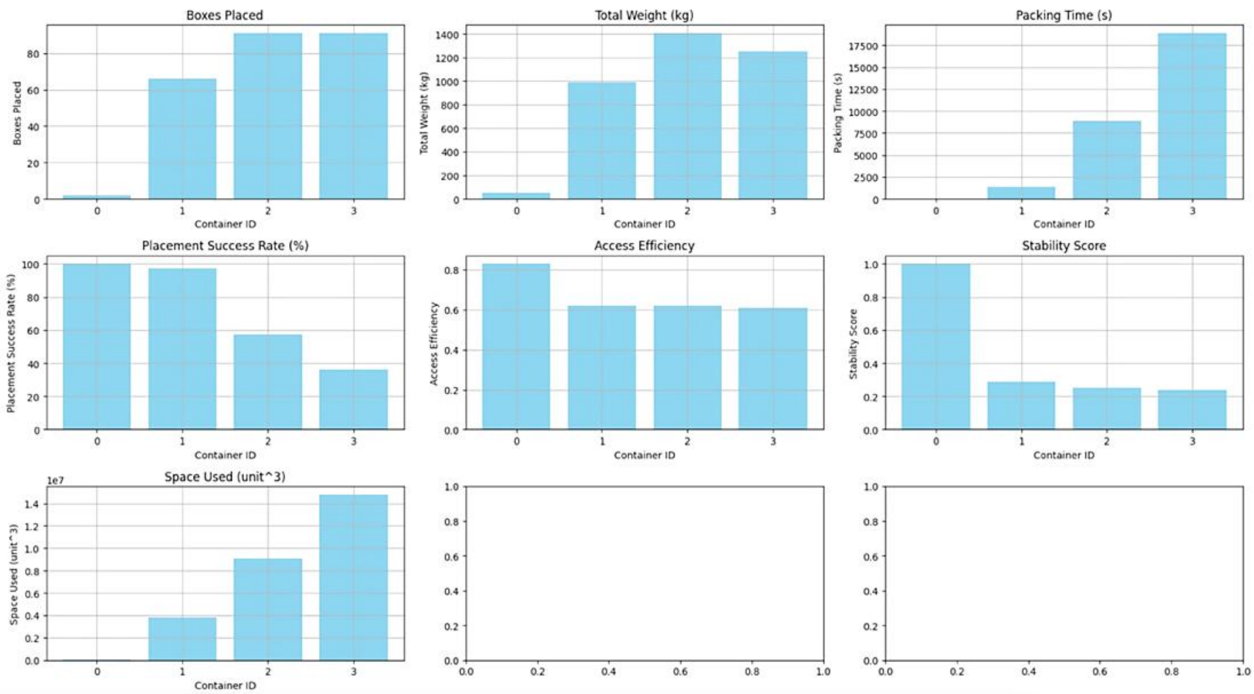


Figure 10. The performance of the Best Fit Algorithm

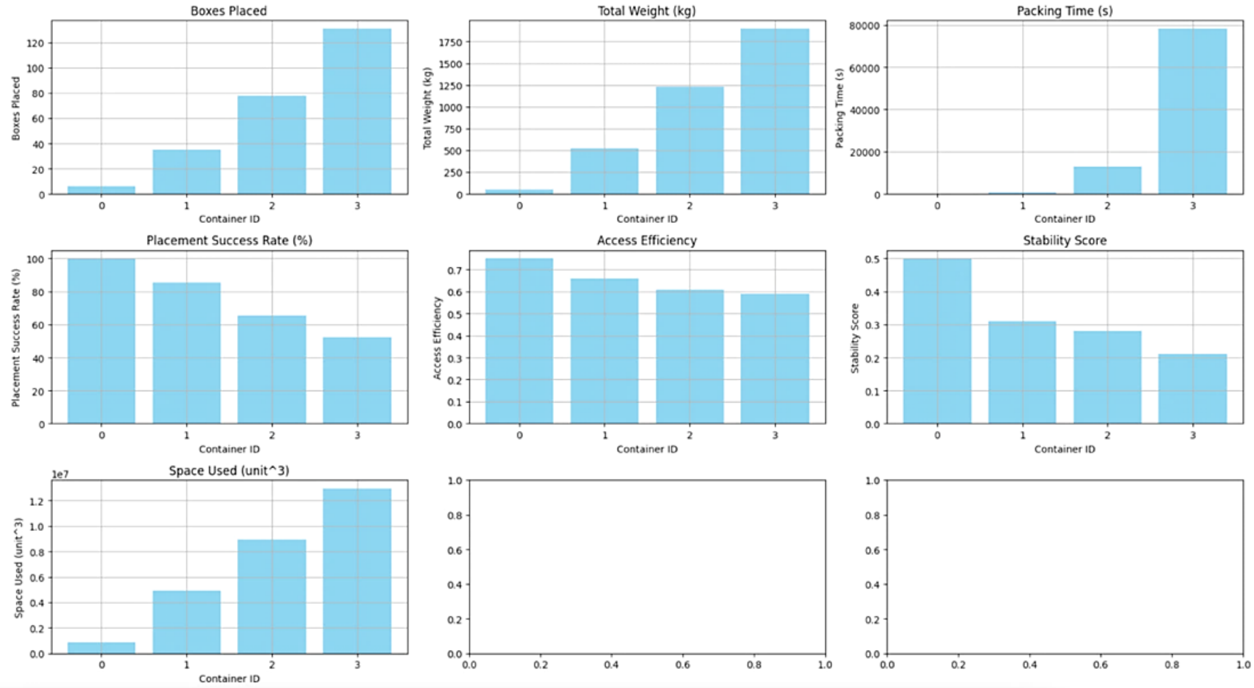


Figure 11. The performance of the Genetic Algorithm

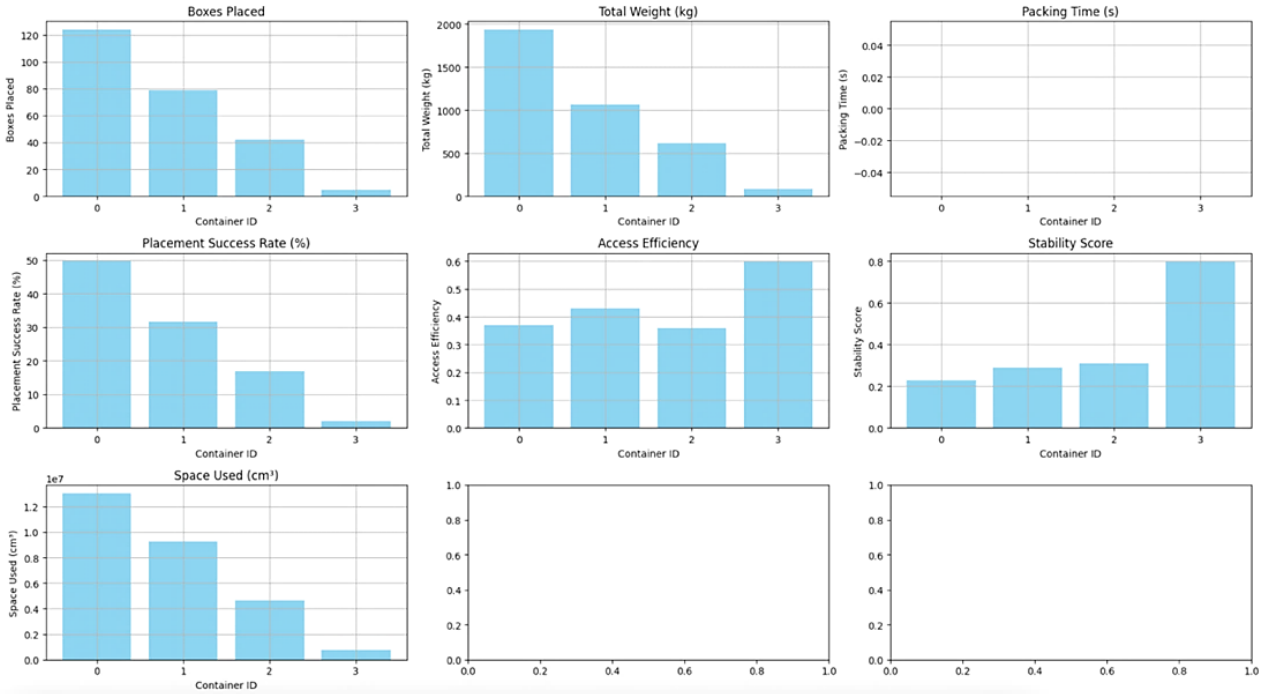


Figure 12. The performance of the Hybrid Algorithm

When we performed metric-by-metric comparison of the four algorithms in Figures 9, 10, 11, and 12, the results show that the genetic algorithm has placed most boxes, especially for packed items the latest box. The other two algorithms, first fit and best fit, demonstrate similar performance with a gradual increase across boxes. The unusual approach is shown by the hybrid algorithm, whereby most boxes are kept in the first container and then reduced afterwards. The hybrid has a suggested approach with early optimization at the first stage.

The performance of the Genetic Algorithm (GA) continues to show good performance for total weight, as it does efficient packing for heavier items. However, GA requires the most computational time compared to the others; this is likely because of the iterative nature of optimization in GA. In terms of stability scores and space used, two algorithms perform similarly good, they are the First fit and GA algorithms. The hybrid algorithm with an unusual approach has successfully utilized spaces in the initial containers and only spent most time in the initial early containers.

Generally, the First Fit and Best Fit methods are beneficial for fast, heuristic-based packing but fall short in efficiency and stability. The Genetic Algorithm excels in optimization but is computationally expensive. The Hybrid PPO algorithm combines the best of both worlds—smart placement, strong stability, and reasonable execution time—making it the most robust and scalable solution for real-world 3D bin packing challenges.

Based on the above experiments, we recommend that the first-fit algorithm is optimal when speed and simplicity are the key points. If the key points are space optimization, the best fit may be a good choice with refinement. The GA lead for maximum packing quality and learning ability. For those who are looking for a balanced real-world approach, the Hybrid algorithm is a smart choice, especially with further tuning.

5. Conclusion

The comparative analysis of the four 3D bin packing algorithms—First Fit, Best Fit, Genetic Algorithm, and Hybrid PPO—highlights the trade-offs between speed, packing efficiency, and intelligent decision-making in logistics optimization. Each algorithm exhibits unique strengths depending on the context of the target application.

The First Fit algorithm proves to be the fastest, making it suitable for scenarios where speed outweighs optimization. However, its lower access efficiency and stability suggest limited effectiveness for complex packing scenarios. The Best Fit method improves space and weight distribution, offering moderate performance gains, but still suffers from inefficiencies in placement and consistency, particularly in smaller containers.

The Genetic Algorithm stands out for its high-quality packing and optimization capabilities, showing excellent box placement and space utilization performance. However, its computational cost is significantly higher, making it more appropriate for offline or non-real-time environments.

The Hybrid PPO algorithm delivers the best overall performance. It achieves a balance between intelligent, stable packing and operational speed. With high placement success rates, enhanced access efficiency, and superior stability scores, it is the most practical and scalable solution for real-world dynamic bin packing tasks, particularly in smart logistics, e-commerce fulfilment, and automated warehousing systems.

These findings support the growing potential of reinforcement learning–based hybrid models in addressing the complexities of real-time, efficient 3D load optimization.

6. Declarations

6.1. Author Contributions

Conceptualization, T.B.K. and D.A.D.; methodology, T.B.K.; software, T.B.K. and H.M.K.; validation, T.B.K., D.A.D., and M.; formal analysis, T.B.K.; investigation, T.B.K. and M.; resources, D.A.D. and M.; data curation, T.B.K.; writing—original draft preparation, T.B.K.; writing—review and editing, D.A.D. and H.M.K.; visualization, T.B.K.; supervision, D.A.D.; project administration, D.A.D.; funding acquisition, D.A.D. All authors have read and agreed to the published version of the manuscript.

6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

6.4. Institutional Review Board Statement

Not applicable.

6.5. Informed Consent Statement

Not applicable.

6.6. Declaration of Competing Interest

The authors declare that there are no conflicts of interest concerning the publication of this manuscript. Furthermore, all ethical considerations, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

7. References

[1] Kmiecik, M., & Wierzbicka, A. (2024). Enhancing Smart Cities through Third-Party Logistics: Predicting Delivery Intensity. Smart Cities, 7(1), 541–565. doi:10.3390/smartcities7010022.

[2] Gupta, H., Shreshth, K., Kharub, M., & Kumar, A. (2024). Strategies to overcome challenges to smart sustainable logistics: a Bayesian-based group decision-making approach. Environment, Development and Sustainability, 26(5), 11743–11770. doi:10.1007/s10668-023-03477-6.

[3] Zhou, F., Yu, K., Xie, W., Lyu, J., Zheng, Z., & Zhou, S. (2024). Digital Twin-Enabled Smart Maritime Logistics Management in the Context of Industry 5.0. IEEE Access, 12, 10920–10931. doi:10.1109/ACCESS.2024.3354838.

[4] Saadian, F., Motameni, H., & Golsorkhtabaramiri, M. (2023). Deadline-aware multi-objective IoT services placement optimization in fog environment using parallel FFD-genetic algorithm. Pervasive and Mobile Computing, 92, 101800. doi:10.1016/j.pmcj.2023.101800.

[5] Mesquita, A. C. P., & Sanches, C. A. A. (2024). Air cargo load and route planning in pickup and delivery operations. Expert Systems with Applications, 249, 123711. doi:10.1016/j.eswa.2024.123711.

[6] Lopes, R., Trovati, M., & Pereira, E. (2024). Volumetric Techniques for Product Routing and Loading Optimisation in Industry 4.0: A Review. Future Internet, 16(2), 39. doi:10.3390/fi16020039.

[7] Verma, R., Singhal, A., Khadilkar, H., Basumatary, A., Nayak, S., Singh, H. V., ... & Sinha, R. (2020). A generalized reinforcement learning algorithm for online 3d bin-packing. arXiv preprint, arXiv:2007.00463.

[8] Yang, S., Song, S., Chu, S., Song, R., Cheng, J., Li, Y., & Zhang, W. (2024). Heuristics Integrated Deep Reinforcement Learning for Online 3D Bin Packing. IEEE Transactions on Automation Science and Engineering, 21(1), 939–950. doi:10.1109/TASE.2023.3235742.

[9] Zhang, C., Wu, Y., Ma, Y., Song, W., Le, Z., Cao, Z., & Zhang, J. (2023). A review on learning to solve combinatorial optimisation problems in manufacturing. IET Collaborative Intelligent Manufacturing, 5(1), 12072. doi:10.1049/cim2.12072.

[10] Cai, Q., Hang, W., Mirhoseini, A., Tucker, G., Wang, J., & Wei, W. (2019). Reinforcement learning driven heuristic optimization. arXiv preprint, arXiv:1906.06639.

[11] Zhang, J., & Shuai, T. (2024). Online Three-Dimensional Bin Packing: A DRL Algorithm with the Buffer Zone. Foundations of Computing and Decision Sciences, 49(1), 63–74. doi:10.2478/fcds-2024-0005.

[12] Gao, C., Zheng, Y., Li, N., Li, Y., Qin, Y., Piao, J., Quan, Y., Chang, J., Jin, D., He, X., & Li, Y. (2023). A Survey of Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions. ACM Transactions on Recommender Systems, 1(1), 1–51. doi:10.1145/3568022.

[13] Spanoudakis, N. I., Akasiadis, C., Iatrakis, G., & Chalkiadakis, G. (2023). Engineering IoT-Based Open MAS for Large-Scale V2G/G2V †. Systems, 11(3), 157. doi:10.3390/systems11030157.

[14] Del Rio, A., Jimenez, D., & Serrano, J. (2024). Comparative Analysis of A3C and PPO Algorithms in Reinforcement Learning: A Survey on General Environments. In IEEE Access. IEEE Access. doi:10.1109/ACCESS.2024.3472473.

[15] Liu, C., & Liu, G. (2024). JointPPO: Diving deeper into the effectiveness of PPO in multi-agent reinforcement learning. arXiv preprint, arXiv:2404.11831.

[16] Kaur, A., & Kaur, B. (2022). Load balancing optimization based on hybrid Heuristic-Metaheuristic techniques in cloud environment. Journal of King Saud University - Computer and Information Sciences, 34(3), 813–824. doi:10.1016/j.jksuci.2019.02.010.

[17] Zhang, W., Maleki, A., & Rosen, M. A. (2019). A heuristic-based approach for optimizing a small independent solar and wind hybrid power scheme incorporating load forecasting. Journal of Cleaner Production, 241, 117920. doi:10.1016/j.jclepro.2019.117920.

[18] Maia, A. M., Ghamri-Doudane, Y., Vieira, D., & Franklin de Castro, M. (2021). An improved multi-objective genetic algorithm with heuristic initialization for service placement and load distribution in edge computing. Computer Networks, 194, 108146. doi:10.1016/j.comnet.2021.108146.

[19] Annie Poornima Princess, G., & Radhamani, A. S. (2021). A Hybrid Meta-Heuristic for Optimal Load Balancing in Cloud Computing. Journal of Grid Computing, 19(2), 21. doi:10.1007/s10723-021-09560-4.

[20] Yan, Y., Liu, G., & Liu, L. (2025). Enhancing Efficiency and Resilience Multiobjective Optimization Model and Algorithm for Site Selection of Emergency Material Storage Sites. KSII Transactions on Internet and Information Systems, 19(2), 344–367. doi:10.3837/tiis.2025.02.001.