Residual Soft Actor-Critic for Fixed-Capacity Retail Replenishment with Two Weekly Deliveries

Hao Trieu Quoc [1*], Duy Nguyen Van Anh [1†] and Thu Nguyen Thi Minh [1†]

[1*]Department of Artificial Intelligence, FPT University, 7 D1 Street, Tang Nhon Phu, Ho Chi Minh City, 700000, Vietnam.

*Corresponding author(s). E-mail(s): haotqse180011@fpt.edu.vn;
Contributing authors: duynvase181823@fpt.edu.vn; thuntmse185043@fpt.edu.vn;
†These authors contributed equally to this work.

Abstract

Retail replenishment under a hard floor-space limit is both a sequential control problem and a physical throughput problem. In the studied five-category consumer-electronics store, a representative seven-day demand volume requires approximately 50.39 m^2 of goods, whereas the available retail area is fixed at 25 m^2. A single weekly delivery is therefore structurally unable to provide full-week coverage. This paper formulates the system as a 26-dimensional-state, five-dimensional-action Markov Decision Process and introduces two replenishment events, on days 0 and 3, so that mid-week sales release capacity for a second delivery. Control is provided by Residual Soft Actor-Critic with automatic entropy-temperature tuning (Residual SAC-AutoAlpha), which blends a service-coverage baseline with a learned continuous actor. The coverage factor and residual weight are selected on distinct validation seeds, checkpoint selection uses three further seeds, and the final 52-week evaluation is conducted on held-out seed 42. In that terminal scenario, the policy attains a 97.81% aggregate fill rate, a 96.46% minimum per-SKU fill rate, mean and P95 weekly shortages of 4.75 and 11.30 units, mean end-of-week inventory of 1.60 demand-days, and accounting profit of 72.216 billion VND while respecting the 25 m^2 hard constraint. Two of 52 weeks nevertheless exceed the 30-unit shortage threshold, with a maximum of 121 units. The findings therefore demonstrate a service-feasible operating point for the reported holdout scenario, rather than a multi-seed guarantee or elimination of tail risk.

Keywords: Residual reinforcement learning, Soft Actor-Critic, Inventory management, Retail replenishment, Capacity constraint, Service level

1 Introduction

Retail inventory control must balance product availability against procurement, holding, and shortage costs. The problem becomes more difficult when heterogeneous products compete for one hard physical capacity and demand changes with weekdays, seasons, and commercial events. Reinforcement learning (RL) offers a natural formulation because replenishment decisions affect both current service and the inventory available in future periods [1]. Continuous-control methods such as Soft Actor-Critic (SAC) are especially relevant when several order quantities must be selected jointly [2, 3].

The system studied here represents a single consumer-electronics store in Vietnam managing five product categories: iPhone, iPad, MacBook, Apple Watch, and AirPods. The store has a non-negotiable floor-space capacity of 25 m^2. At the measured demand scale, approximately 50.39 m^2 of incoming goods would be required to cover a representative seven-day period. This mismatch cannot be corrected by longer training or reward tuning because it is a physical throughput limitation. The operational design must first create another opportunity to replenish after sales release occupied space.

Accordingly, this work uses two delivery events per week, on days 0 and 3. The same five-dimensional action is interpreted as a per-delivery replenishment proposal and is passed through the capacity constraint at each event. On top of this structural redesign, the controller uses a service-oriented baseline as an action prior and learns a bounded residual with SAC. This separation assigns distinct roles to engineering and learning: the delivery schedule addresses physical feasibility, the baseline keeps decisions near a service-feasible region, and the learned component adjusts the joint five-product action.

The contributions of this paper are:

- a fixed-capacity weekly retail MDP with two within-week replenishment events and an auditable requested-to-executed action transformation;
- a Residual SAC-AutoAlpha policy that combines a demand-coverage prior with a learned continuous correction, including the required log-probability adjustment;
- a seed-separated selection procedure for the coverage factor, residual weight, and training checkpoint; and
- an evaluation that jointly reports service, tail shortage, inventory, profit, and hard-capacity compliance while retaining the limitations of a single terminal holdout scenario.

Section 2 reviews relevant inventory and RL work. Section 3 defines the MDP, delivery mechanism, reward, and residual policy. Section 4 describes validation and evaluation. Sections 5 and 6 present and interpret the findings, followed by limitations in Section 7 and conclusions in Section 8.

2 Related Work

Classical replenishment policies provide interpretable rules for balancing order and holding costs, but a shared hard-capacity constraint couples the five product decisions and makes independently tuned rules less reliable. The coupling is operational: requesting more of one product reduces the floor space available to every other product in the same delivery.

RL expresses this interaction as an MDP and optimizes discounted return under stochastic transitions [1]. Deep RL has been applied to inventory problems in which nonlinear costs and constraints make fixed policies difficult to tune [4]. The present work is methodologically motivated by the SAC-based replenishment formulation of Zhang, He, and Zheng [5], but it considers a single-echelon, five-category retail store with a shared floor-space constraint rather than their multi-echelon FMCG inventory system.

SAC learns a stochastic continuous policy using an entropy-regularized objective and twin critics [2]. Automatic temperature tuning adapts the entropy coefficient instead of fixing it manually [3]. In this paper, "AutoAlpha" refers only to this learned entropy temperature; the actor, critic, and temperature optimizers all retain fixed learning rates. The deployed policy is not vanilla SAC: it is a residual policy built around an explicit service-coverage prior. This naming distinction is material because the prior supplies 90% of the final blended proposal at the selected operating point.

3 Methodology

3.1 Weekly Markov Decision Process

The replenishment problem is represented by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where one decision epoch corresponds to one seven-day period and $\gamma = 0.99$. The transition combines the deterministic delivery and capacity rules with stochastic daily demand. Figure 1 summarizes the closed loop.

The state is a bounded continuous vector $s_t \in \mathbb{R}^{26}$:

$$s_t = [I_t, \widehat{D}_t, C^{\text{buy}}, C^{\text{hold}}, C^{\text{short}}, d_t], \tag{1}$$

where each of the first five blocks contains one normalized value per SKU: current inventory I_t, seven-day demand forecast $\widehat{D}_t$, purchase cost, holding cost, and shortage penalty. The final scalar d_t is the normalized time-of-year index. Thus, the model has a 26-dimensional state vector rather than 26 discrete states. The forecast is a noisy sample from the demand generator and is not the realized future demand.

The policy proposal $a_t^{\text{final}} \in \mathbb{R}^5$ specifies a continuous per-delivery order quantity for each SKU. At each delivery the environment applies the following pipeline: NumPy rounding to integers; clipping to $[0, a_i^{\max}]$; clipping to the remaining per-SKU capacity; and, only when the resulting footprint would

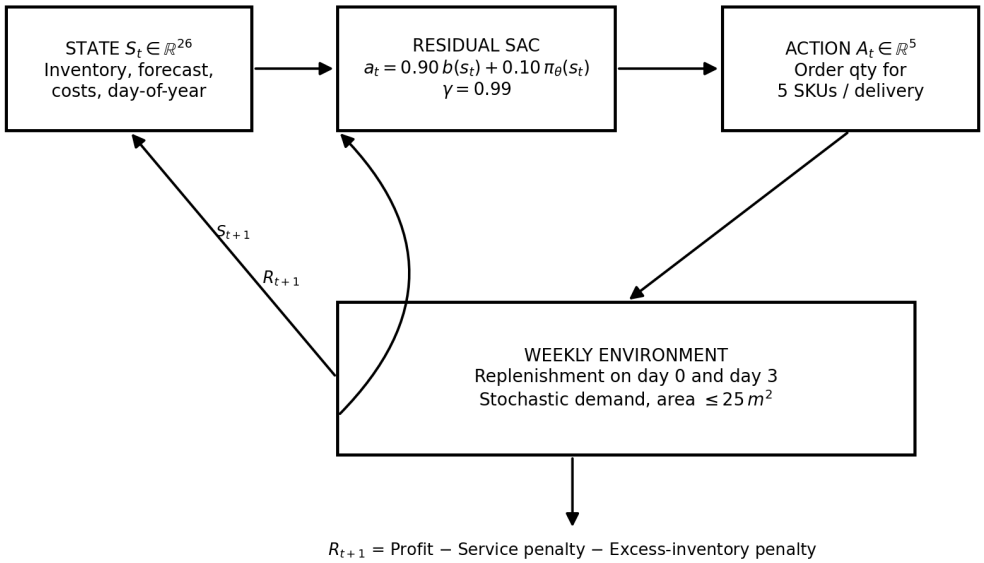

Figure 1. MDP architecture of the fixed-capacity inventory control system.

Fig. 1 MDP architecture of the fixed-capacity inventory system. The Residual SAC-AutoAlpha controller maps the observed state to a five-product replenishment proposal; the environment enforces the 25 m^2 constraint at day-0 and day-3 deliveries, simulates daily demand, and returns the weekly reward and next state.

overflow the store, uniform area scaling followed by flooring. Let q_i denote the integer proposal after the first three operations, f_i the footprint of SKU i, I_i inventory immediately before that delivery, $D = \sum_i f_i q_i$ requested delivery area, and $C_{\text{rem}} = \max(25 - \sum_i f_i I_i, 0)$. The implementation-safe scale is

$$\kappa = \begin{cases} 1, & D \leq 0, \\ \min(1, C_{\text{rem}}/D), & D > 0, \end{cases} \qquad a_i^{\text{exec}} = \lfloor \kappa q_i \rfloor \,. \tag{2}$$

Only executed units enter inventory and incur procurement cost. Because one common κ scales the full vector, the constraint explicitly couples all SKUs.

3.2 Two-Delivery Replenishment

A representative week requires approximately 50.39 m^2 of goods for seven-day demand coverage, exceeding the 25 m^2 store capacity. The policy and baseline are evaluated once at the start of the weekly step, and the resulting continuous proposal is reused on day 0 and day 3. The environment does not recompute the baseline or call the actor at mid-week. It does, however, update inventory after three days of sales and independently repeats rounding, product-capacity clipping, area scaling, and integer execution for the second delivery. Procurement cost is charged separately from the units actually executed at each event. This mechanism increases weekly throughput without relaxing the physical limit or changing the actor output from five to ten dimensions.

Reusing the action preserves a compact and interpretable policy: the output is the replenishment level requested on each trip. It also imposes a limitation, because the policy cannot choose different quantities for the two delivery events. A ten-dimensional extension with independently controlled deliveries is left for future work.

3.3 Accounting Profit and Training Reward

Accounting profit is reported independently of reward shaping:

$$\Pi_t = \text{Revenue}_t - \text{Procurement}_t - \text{Holding}_t - \text{ShortageCost}_t. \tag{3}$$

The learning signal is

$$r_t = 10^{-8}\big(\Pi_t + \lambda_m B_t - P_t^{\text{tail}} - P_t^{\text{fill}} - P_t^{\text{excess}}\big), \tag{4}$$

where P_t^{tail} penalizes aggregate shortage beyond the weekly target, P_t^{fill} penalizes per-SKU fill-rate gaps, and P_t^{excess} penalizes end inventory beyond a two-day demand target. The runtime margin-bonus weight is $\lambda_m = 0$; the term is retained only as an ablation control. The shaping penalties and scale factor are optimization devices, not business expenses, and are never subtracted again from reported accounting profit.

The implementation defines these terms explicitly. Let L_t be aggregate unmet units, $L_{i,t}$ per-SKU unmet units, $D_{i,t}$ realized per-SKU demand, $I_{i,t}^{\text{end}}$ end inventory, $\bar{C}^{\text{buy}}$ mean purchase cost, and μ_i the configured normal daily demand. With $[x]_+ = \max(x, 0)$,

$$P_t^{\text{tail}} = \bar{C}^{\text{buy}} \rho_{\text{tail}} \frac{[L_t - L_0]_+^2}{L_0}, \qquad L_0 = 30, \quad \rho_{\text{tail}} = 0.10, \tag{5}$$

$$u_{i,t} = (1 - F_{\min}) D_{i,t}, \qquad F_{\min} = 0.93, \tag{6}$$

$$P_t^{\text{fill}} = \sum_i C_i^{\text{buy}} \rho_{\text{fill}} \frac{[L_{i,t} - u_{i,t}]_+^2}{\max(u_{i,t}, 1)}, \qquad \rho_{\text{fill}} = 0.10, \tag{7}$$

$$P_t^{\text{excess}} = \sum_i C_i^{\text{buy}} \rho_{\text{inv}} \frac{[I_{i,t}^{\text{end}} - 2\mu_i]_+^2}{\max(2\mu_i, 1)}, \qquad \rho_{\text{inv}} = 0.05. \tag{8}$$

3.4 Residual SAC-AutoAlpha

The service baseline converts forecast and inventory into a per-delivery target:

$$b(s) = \text{clip}\left(\frac{c\widehat{D} - I}{2}, 0, a^{\max}\right), \tag{9}$$

where c is the coverage factor and division by two accounts for the two delivery events. The final proposal blends this prior with the SAC actor:

$$a^{\text{final}}(s) = (1 - w) b(s) + w \pi_\theta(s). \tag{10}$$

The selected values are $c = 1.20$ and $w = 0.10$. Both components use the same per-SKU action scale $[0, a_i^{\max}]$, so the 90/10 coefficients are directly interpretable. The actor therefore remains active but makes a bounded correction around the service prior. Since Eq. (10) is affine in the actor output, the implementation adjusts the sampled log-density as

$$\log \pi_{\text{final}}(a \mid s) = \log \pi_{\text{actor}}(a \mid s) - 5 \log w. \tag{11}$$

The same corrected value in Eq. (11) is used in the Bellman target, actor loss, and temperature loss. The actor's base log-density includes the Gaussian and tanh Jacobian terms but omits the constant Jacobian of rescaling from $[-1, 1]$ to the heterogeneous quantity bounds. Accordingly, Eq. (11) describes the implemented pre-environment continuous density, not a density after the non-invertible rounding and capacity projection.

The replay buffer stores $(s_t, a_t^{\text{final}}, r_t, s_{t+1}, d_t)$, where a_t^{final} is the blended continuous proposal before any environment-side projection. The critics are trained on this stored proposal and on newly sampled blended proposals. The environment, by contrast, executes a_t^{exec} from Eq. (2). This preserves consistency between the policy density and the critic's declared continuous action space, but the reward is generated after a many-to-one projection; the resulting action aliasing is stated as a limitation in Section 7.

For completeness, the implemented SAC objectives are

$$y_t = r_t + (1 - d_t)\gamma \left[\min_{j \in \{1,2\}} Q_{\bar{\phi}_j}(s_{t+1}, a'_{t+1}) - \alpha \log \pi_{\text{final}}(a'_{t+1} \mid s_{t+1})\right], \tag{12}$$

$$J_Q = \sum_{j=1}^{2} \mathbb{E}\left[(Q_{\phi_j}(s_t, a_t^{\text{final}}) - y_t)^2\right], \tag{13}$$

$$J_\pi = \mathbb{E}\left[\alpha \log \pi_{\text{final}}(a'_t \mid s_t) - \min_j Q_{\phi_j}(s_t, a'_t)\right], \tag{14}$$

$$J_\alpha = -\mathbb{E}\left[\log \alpha \left(\log \pi_{\text{final}}(a'_t \mid s_t) + \mathcal{H}_{\text{target}}\right)\right]. \tag{15}$$

The actor accepts 26 inputs, uses two 256-unit ReLU hidden layers, and produces the mean and log standard deviation of a five-dimensional Gaussian before tanh squashing and action rescaling. Two critics accept the joint 31-dimensional state–action input, and the minimum of their estimates is used to reduce overestimation. Target critics are updated with $\tau = 0.005$. The temperature starts at $\alpha = 1$ and is learned with the implemented target $\mathcal{H}_{\text{target}} = -5$; actor, critic, and temperature optimizers use Adam with fixed learning rate 3×10^{-4}. Appendix A lists the remaining implementation parameters and normalization constants.

4 Experimental Setup

4.1 Hyperparameter Selection

Coverage and residual weight were selected in two stages using disjoint validation pools. First, $c \in \{0.8, 1.0, 1.2, 1.4, 1.6\}$ was evaluated on seeds 6201–6205 before introducing a learned residual. A candidate was eligible only if aggregate fill rate was at least 97% and every per-SKU fill rate cleared its warning floor. Coverage $c = 1.20$ was selected as the least-inventory eligible candidate, as shown in Figure 2.

Figure 2. Service-inventory frontier under the fixed 25 m^2 constraint

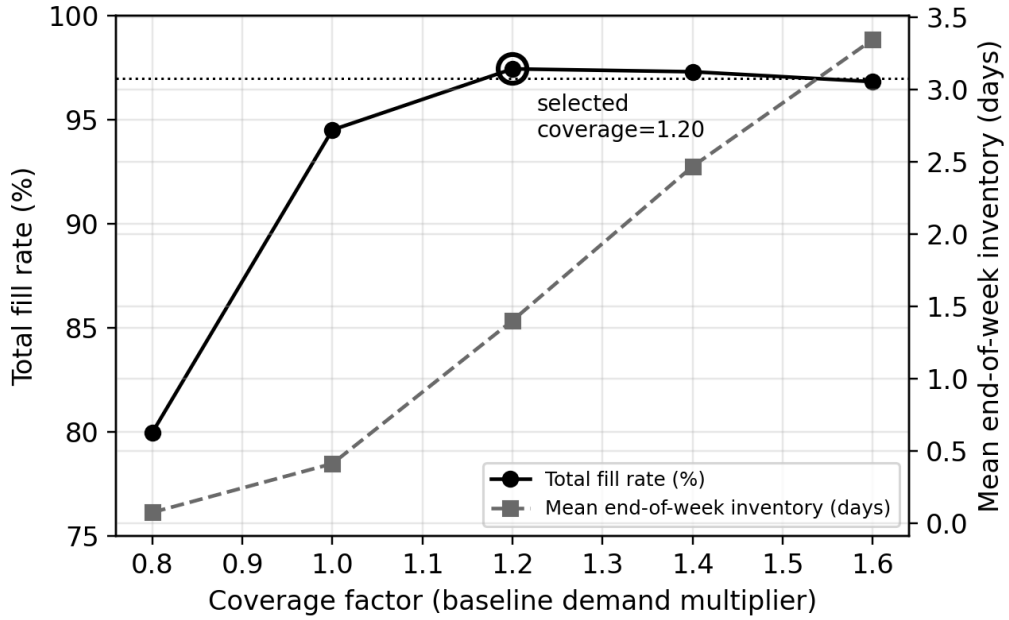

Fig. 2 Service–inventory frontier under the hard 25 m^2 constraint. Coverage 1.20 is the least-inventory tested setting that clears all service targets on validation seeds 6201–6205.

The actor was trained with service projection enabled and $w = 0.10$, the runtime value in the configuration. Second, after checkpoint selection, this actor was frozen and $w \in \{0, 0.05, 0.075, 0.10, 0.20\}$ was varied only during deterministic rollout on seeds 7601–7605. No candidate-specific retraining and no retraining after selection were performed. Only candidates that passed the complete service and inventory gate were compared by accounting profit; $w = 0.10$ was selected. Figure 3 therefore reports a post-training frozen-actor ablation, not full hyperparameter optimization of separately trained policies. Both selected values are empirical choices from finite seed sweeps rather than theoretical constants or values imported from retail literature. The $w = 0$ rollout is safe because deterministic selection uses only the blend and never evaluates $\log w$; stochastic training with $w = 0$ is not supported by the implementation.

Figure 3. Residual-policy-weight ablation (Actor frozen, coverage=1.20)

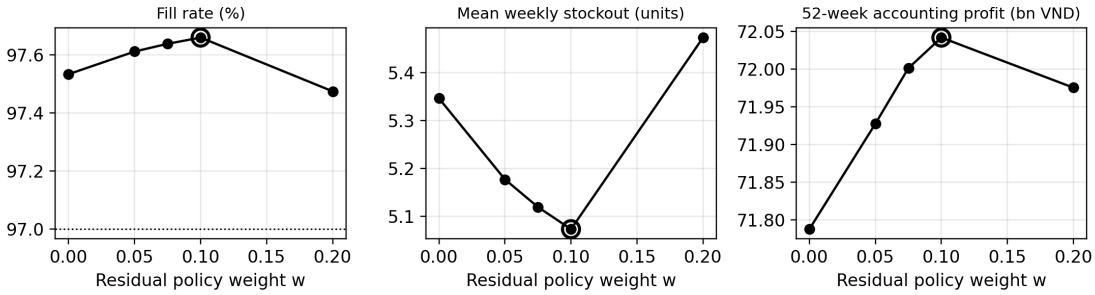

Fig. 3 Residual-weight ablation with the actor frozen at coverage 1.20. The selected value $w = 0.10$ is the profit-preferred candidate among settings that passed the joint service and inventory gate on validation seeds 7601–7605.

4.2 Training, Selection, and Terminal Evaluation

The current pipeline separates the seed roles summarized in Table 1. Training uses seed 7201. Every ten episodes, checkpoint candidates are evaluated on seeds 7301–7303. Eligibility requires aggregate fill rate of at least 97%, every per-SKU fill rate of at least 93%, mean weekly shortage no greater than 30 units, and mean end-of-week inventory no greater than two demand-days. Candidates are ranked lexicographically by eligibility, aggregate fill rate, minimum per-SKU fill rate, negative P95 shortage, and profit. Thus, profit is the final tie-breaker rather than the primary checkpoint objective. The final reported rollout uses seed 42, which is not used for gradient updates or checkpoint selection.

Table 1 Separation of random-seed roles in the current pipeline.

Stage	Seeds	Purpose
Coverage validation	6201–6205	Select service coverage factor
Residual validation	7601–7605	Select residual weight
Training	7201	Perform gradient updates
Checkpoint selection	7301–7303	Apply the release gate
Terminal evaluation	42	Produce reported 52-week metrics

The environment represents 365 days, which is not divisible by seven. Evaluation is therefore truncated to 52 complete weeks (364 days), avoiding a partial 53rd decision period. The terminal rollout reports aggregate and per-SKU fill rate, weekly shortage distribution, end-of-week inventory in demand-days, accounting profit, and post-delivery footprint. Monetary results are simulated VND values rather than audited observations from an operating retailer.

5 Results

Table 2 reports the terminal 52-week evaluation on seed 42. The policy clears the aggregate service target with a fill rate of 97.81%, while the lowest product-specific fill rate is 96.46%. Mean weekly shortage is 4.75 units and the P95 is 11.30 units. Mean end-of-week inventory remains below the two-day gate at 1.60 demand-days. The maximum observed post-delivery footprint is 24.79 m^2, with zero violations of the 25 m^2 limit across the 52 weekly rollouts.

The distributional metrics qualify the favorable averages. Two weeks exceed the 30-unit shortage threshold and the maximum weekly shortage reaches 121 units. Thus, the reported mean and P95 targets are met, but the policy does not guarantee that every week clears the operational threshold. The 72.216 billion VND profit is accounting profit from Eq. (3); it excludes the artificial shaping penalties in Eq. (4).

6 Discussion

The results indicate that the combined design reaches a useful operating point in the held-out scenario: high aggregate and per-product service, low typical shortage, less than two days of end inventory,

Table 2 Terminal evaluation of Residual SAC-AutoAlpha over 52 complete weeks (seed 42).

Metric	Result
Mean weekly shortage	4.75 units
Total 52-week shortage	247 units
P95 weekly shortage	11.30 units
Maximum weekly shortage	121 units
Weeks exceeding 30-unit shortage target	2/52
Aggregate fill rate	97.81%
Lowest per-SKU fill rate	96.46%
Mean end-of-week inventory	1.60 demand-days
52-week accounting profit	72.216 billion VND
Peak footprint after a delivery	24.79 m^2
Capacity violations	0

positive accounting profit, and strict capacity compliance. These outcomes should be attributed to the complete design rather than to the learned residual alone. Two deliveries resolve the single-trip throughput mismatch, the baseline provides most of the service-oriented action, and SAC adjusts the joint proposal within the remaining 10% residual weight.

The residual construction also changes the interpretation of learning performance. The policy is neither a fixed base-stock rule nor vanilla SAC. A residual weight of 0.10 means the hand-designed prior materially constrains the operating region, while the actor still changes every proposal and is trained with twin critics and entropy regularization. The residual-weight ablation establishes the selected operating point within the tested grid, but it does not isolate all possible interactions between architecture, training duration, and the service prior.

Tail behavior remains operationally important. Although P95 shortage is 11.30 units, the maximum is 121 units. In extreme weeks, demand can exceed the quantity that two deliveries can physically move through 25 m^2. Reporting only the mean would hide this risk. A practical deployment would require monitoring, exception handling, or an additional logistics option for peak periods.

7 Limitations and Future Work

The evaluation has the following limitations:

- **Single terminal scenario.** Seed roles are separated, but headline metrics are produced by one terminal seed. They provide no multi-seed variance, confidence interval, or statistical guarantee.
- **One action reused twice.** The same five-dimensional vector is executed on days 0 and 3. A ten-dimensional policy could condition the two deliveries independently, at the cost of a larger action space and additional training burden.
- **Empirical control constants.** Coverage 1.20 and residual weight 0.10 are selected from five-seed grids. They are not analytical optima and may not transfer unchanged to another store or demand distribution.
- **Frozen-actor weight ablation.** The actor was trained with $w = 0.10$; alternative residual weights were evaluated after training without candidate-specific retraining. The sweep measures deployment-time sensitivity of one checkpoint and does not establish which weight would be optimal if each candidate were trained end to end.
- **No pure-policy attribution.** Because the deployed policy is a 90/10 blend, the results cannot be presented as the achievement of an unconstrained SAC actor. Future experiments should compare the residual design with a tuned rule-based policy and matched vanilla continuous-control baselines.
- **Entropy-space calibration.** The implementation applies the residual Jacobian correction to the critic target, actor loss, and alpha loss while retaining target entropy -5. It also omits the constant Jacobian of heterogeneous action-range rescaling. Consequently, the numerical entropy target is not separately calibrated for the final blended quantity space. A future implementation should either tune alpha in actor space or derive a fully consistent final-space target and density.
- **Continuous proposal versus executed action.** The replay buffer and critics use the pre-projection blended proposal, whereas reward follows rounded, clipped, capacity-scaled integer

execution. Multiple proposals can therefore induce the same transition and reward. Future instrumentation should expose requested action, scale factor, and executed action separately and quantify this aliasing.

- **Noisy forecast.** The seven-day forecast and realized demand are distinct samples from the same stochastic generator. The forecast is informative but is not ground truth.
- **Economic-input provenance.** Purchase costs, carrying-cost rates, and shortage penalties include market benchmarks and modeling assumptions rather than audited invoices. Profit should therefore be interpreted as simulated scenario profit.
- **Residual tail risk.** Two of 52 weeks exceed the shortage target. The design optimizes a service–inventory–profit compromise under a hard constraint; it does not eliminate stockout and excess-inventory risk simultaneously.

Future work should repeat terminal evaluation over a locked multi-seed test pool, report uncertainty intervals, train matched coverage-only, vanilla SAC-AutoAlpha, and fixed-alpha residual baselines, evaluate independent day-0 and day-3 actions, and test additional logistics capacity for extreme demand weeks. These extensions should preserve the separation between validation seeds and terminal reporting.

8 Conclusion

This paper presented a fixed-capacity retail replenishment system combining a structural two-delivery schedule with Residual SAC-AutoAlpha. The two delivery events address a physical mismatch between approximately 50.39 m^2 of representative weekly coverage and the 25 m^2 store limit, while the residual policy adjusts a service-oriented baseline using a joint continuous actor. In the held-out 52-week seed-42 scenario, the system achieves a 97.81% aggregate fill rate, a 96.46% minimum per-SKU fill rate, 4.75 mean weekly shortage units, 11.30 P95 shortage units, 1.60 demand-days of end inventory, and 72.216 billion VND accounting profit without violating the area constraint. Because two weeks still exceed the shortage threshold and the terminal evidence comes from one seed, the appropriate conclusion is that the method reaches a service-feasible operating point in the reported scenario, not that it removes tail risk or guarantees seed-robust performance.

Appendix A Implementation Parameters

Table A1 reports the settings used by the training pipeline. Unless stated otherwise, optimizer parameters are the PyTorch Adam defaults.

Table A1 Residual SAC-AutoAlpha training and policy configuration.

Parameter	Implemented value
State/action dimensions	26 / 5
Actor and critic hidden layers	$[256, 256]$, ReLU
Actor log-standard-deviation range	$[-20, 2]$
Discount / target update	$\gamma = 0.99$, $\tau = 0.005$
Actor, critic, alpha learning rates	3×10^{-4} each
Adam $(\beta_1, \beta_2, \epsilon)$ / weight decay	$(0.9, 0.999, 10^{-8})$ / 0
Initial alpha / target entropy	1 / -5
Gradient clipping	actor and critic norm 1.0
Replay capacity / mini-batch	100,000 / 256
Training / warm-up	200 / 10 episodes
Episode and update schedule	52 transitions; 52 updates after warm-up
Target-network update interval	every gradient update
Coverage / residual weight	1.20 / 0.10
Actor retrained after weight sweep	no
Action lower bounds	$[0, 0, 0, 0, 0]$
Action upper bounds	$[50, 35, 20, 35, 100]$
Reward scale / margin-bonus weight	10^{-8} / 0

The normalized state uses $I/[80, 60, 35, 60, 150]$, $\widehat{D}/100$, purchase cost divided by 50 million VND, daily carrying cost divided by 50,000 VND, shortage penalty divided by 1 million VND, and day

divided by 365. No running normalization statistics are fitted, so validation or terminal observations do not update normalization state.

Table A2 Per-SKU environment parameters in product order iPhone, iPad, MacBook, Apple Watch, and AirPods.

Parameter	iPhone	iPad	MacBook	Watch	AirPods
Initial inventory (units)	50	30	15	30	80
Maximum inventory (units)	80	60	35	60	150
Maximum order per delivery	50	35	20	35	100
Footprint (m^2/unit)	0.10	0.20	0.35	0.08	0.04
Purchase cost (million VND)	25	18	35	10	4
Procurement multiplier	1.00	0.90	0.95	0.85	0.80
Normal daily demand	6	4	2	4	10

Selling price is 1.4 times purchase cost for every SKU. Shortage cost is 10% of purchase cost per unmet unit, and daily carrying cost is purchase cost times the assumed 20% annual carrying rate divided by 365. Demand is sampled from a Gaussian with standard deviation $\max(0.15\mu_i, 1)$, then multiplied in its mean by weekday, sinusoidal seasonal, and configured holiday factors before being clipped below at zero and converted to integer units. Lost demand is not backordered.

References

[1] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd edn. MIT Press, Cambridge, MA, USA (2018)

[2] Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International Conference on Machine Learning, pp. 1861–1870 (2018). PMLR

[3] Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., Levine, S.: Soft actor-critic algorithms and applications. arXiv preprint arXiv:1812.05905 (2018)

[4] Gijsbrechts, J., Boute, R.N., Van Mieghem, J.A., Zhang, D.J.: Can deep reinforcement learning improve inventory management? performance on lost sales, dual-sourcing, and multi-echelon problems. Manufacturing & Service Operations Management **24**(3), 1349–1368 (2022)

[5] Zhang, Y., He, L., Zheng, J.: A deep reinforcement learning-based dynamic replenishment approach for multi-echelon inventory considering cost optimization. Electronics **14**(1), 66 (2025) https://doi.org/10.3390/electronics14010066