Physics-Aware Robotic Palletization with Online Masking Inference

Tianqi Zhang, Zheng Wu, Yuxin Chen, Yixiao Wang, Boyuan Liang,
Scott Moura, Masayoshi Tomizuka, Mingyu Ding, Wei Zhan

***Abstract*— The efficient planning of stacking boxes, especially in the online setting where the sequence of item arrivals is unpredictable, remains a critical challenge in modern warehouse and logistics management. Existing solutions often address box size variations, but overlook their intrinsic and physical properties, such as density and rigidity, which are crucial for real-world applications. We use reinforcement learning (RL) to solve this problem by employing action space masking to direct the RL policy towards valid actions. Unlike previous methods that rely on heuristic stability assessments which are difficult to assess in physical scenarios, our framework utilizes online learning to dynamically train the action space mask, eliminating the need for manual heuristic design. Extensive experiments demonstrate that our proposed method outperforms existing state-of-the-arts. Furthermore, we deploy our learned task planner in a real-world robotic palletizer, validating its practical applicability in operational settings.**

I. INTRODUCTION

In modern warehouse and logistics management, stacking boxes continues to be a common challenge. In the past, due to the smaller scale of trade and lower efficiency requirements, workers could rely on their experience to decide how each box should be placed. However, with the globalization of trade, there is a need for fast and stable box stacking, and a good solution for this is robotic palletization [1] [2].

The task planning for robotic palletization in most industrial environments can be conceptually framed as a variant of the online 3D Bin Packing Problems (BPP), where the inventory of items is predetermined, yet their sequence of arrival remains unpredictable [3]. Existing works often focus only on boxes' size variations [4] [5], while they overlook the intrinsic differences between boxes. However, in real logistics scenarios, the overall density and rigidity of boxes can vary significantly. Placing a heavy box on top of a soft and lightweight box can pose substantial risks. Therefore, in this study, we considered the density and rigidity of the boxes to better align with the demands of real-world logistics scenarios. Beyond that, we follow [6] and consider a buffer size, which allows the robot to utilize an auxiliary space for storing up to N pending items, thereby expanding its operational capabilities beyond merely handling the immediate one.

In early studies addressing online 3D bin packing problems, hand-coded heuristic methods were primarily used [7] [8]. With the increasing amount of research related to reinforcement learning (RL) [9], more and more studies are exploring how to use RL to solve this problem [4] [5] [6].

*This work was supported by Berkeley DeepDrive. All authors are with University of California, Berkeley, 94720, USA. The authors thank Anyware Robotics for their hardware support.

Due to the combinatorial nature of the action space of the problem, applying RL to solve online 3D BPP suffers from the problem of large action space, which will complicate the RL training process [10]. One commonly used solution to the problem of RL with large action space is through "invalid action masking" [11] [12], which identifies and masks out the invalid actions and directs the policy to exclusively sample valid actions during the learning phase.

Our work also adopts action space masking, pinpointing valid actions as task plans that guarantee the stability of intermediate item stacks on the pallet under gravitational forces throughout the training process. However, the space masking methods in previous studies are not applicable in the setting of this paper. The heuristic-based methods [13] [14] used in previous work to assess stability become ineffective, as they did not take into account the rigidity and density of the boxes. Although Wu et al. [6] has employed neural networks to evaluate stability, they still require an initial round of data collection using heuristic methods before training the neural network to serve as action masking model.

To address the reliance on heuristic methods, we propose a new framework that uses online learning to train an action space mask, which entirely eliminates the need for manually designing a heuristic method. Specifically, leveraging the image-like characteristics of observation and action mask data, we employ a semantic segmentation paradigm [15], [16] to train the action masking model. The training dataset is compiled through an online data collection phase, utilizing a physics engine to verify the stability of specific placements across various pallet configurations. Moreover, we design the method for collecting and managing online data, allowing the action masking model to efficiently learn the underlying physical principles of box stacking within the RL training loop.

To evaluate the effectiveness of our proposed methodology, we performed a series of comprehensive experiments and compared our results with current RL-based online 3D bin packing solutions. The experimental outcomes reveal that our method significantly surpasses existing baselines. Additionally, we deployed our learned task planner in a real-world robotic palletizer, showcasing the practical applicability of our approach in actual operational environments.

II. RELATED WORKS

Offline 3D bin packing problem. It is a well-known combinatorial optimization problem where the objective is to efficiently pack a set of three-dimensional items into one or more bins (or containers) of fixed dimensions, maximizing

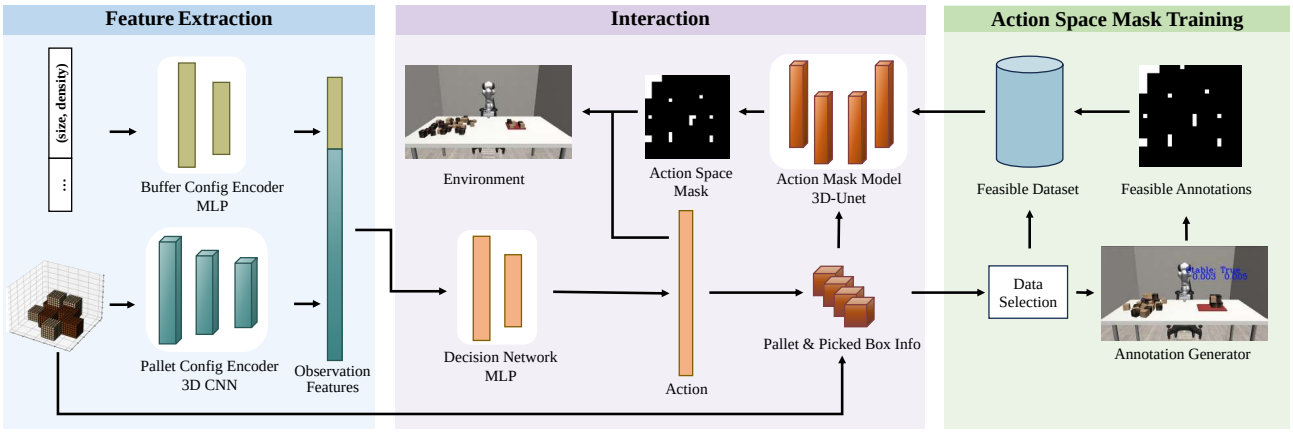

Fig. 1: The whole framework of our proposed method. The entire framework can be divided into three parts. The first part is feature extraction. At each timestep t, given the state $s_t = \{C_t, d_t\}$, we process the pallet configuration C_t using a 3D CNN and the properties of the boxes d_t in the buffer using an MLP. These two components are then concatenated to form the observation feature. The second part is interaction. The latter half of the policy network outputs an action based on the observation feature. We use a 3D-UNet [17] as the action masking model. However, we have added a convolutional layer at the end to transform the multi-channel 3D array input into a single-channel 2D map. It generates an action space mask that filters out unstable box placement points, by using the selected box's properties and the current pallet configuration. The action is then executed in the environment, and the result is obtained. The third part is the training of the action masking model. Unlike the first two parts, which are executed at each timestep, this part only occurs when the policy is updated. First, multiple parallel simulators are used to generate the corresponding feasible annotations for the data selected during the rollout. These data points and their annotations are then appended to the feasible dataset, which is subsequently used to train the action masking model over several epochs.

space utilization. In the offline version, all the items and their dimensions are known in advance before packing begins, which has been particularly challenging due to its NP-hard nature. For smaller instances, methods like integer linear programming (ILP) or branch-and-bound can be applied to find optimal solutions. To deal with a larger number of boxes, early research focuses on heuristic and meta-heuristic methods [14] [18] [19]. More recently, deep reinforcement learning is applied to tackle this task [20] [21].

Online 3D bin packing problem. This is a variation of the 3D BPP where items arrive sequentially over time, requiring placement decisions to be made without knowledge of future items. Unlike the offline version, where all items are known beforehand, this uncertainty necessitates a balance between immediate efficiency and maintaining flexibility for future items, and algorithms must be adaptable to varying patterns and sizes of incoming items. To tackle this problem, researchers also begin with adaptive heuristics algorithms in the early stage. Initially, solutions are based on deep-bottom-left (DBL) heuristics[22] [7], later evolving to the Heightmap-Minimization technique [8]. Nowadays, deep reinforcement learning (DRL) has become popular for addressing this problem, though it is challenged by large action space. Zhao et al. [4] address this challenge by implementing straightforward heuristics to reduce the action space, in another work they [13] seek to alleviate this issue by devising a packing configuration tree that employs more intricate heuristics to identify a subset of feasible actions. Wu et al. [6] first attempt to use neural networks as an action space mask, but it also relies on heuristic methods for data collection.

In this work, we study a variant of the online 3D bin packing problem (BPP). We take the intrinsic properties of the boxes into account, such as density and rigidity. With this consideration, even if a box is fully supported, it will collapse if the box beneath it lacks the rigidity to bear the weight of the box above. As a result, there are fewer stable placement positions for the boxes in our problem compared to the standard online 3D BPP.

Action masking model. One of the major challenges in using RL to solve the online 3D BPP is the large action space. To address this, action space masking is typically employed to reduce the size of the action space. Previous work [4] used heuristic methods as action masking models, often manually adding constraints related to the box's support area and the number of supporting edges, only considering placements that satisfy these constraints as stable. [6] trains a neural network as an action masking model through supervised learning. However, to solve the out-of-distribution problem common in supervised learning, the method in [6] required multiple rounds of training, followed by offline training of the neural network. The first round of RL training still depended on heuristic methods. Therefore, this neural network approach is an improvement over heuristic methods, but it does not eliminate the reliance on them.

III. Method

A. Problem Formulation

The objective of palletization is to maximize space utilization while ensuring stability. Space utilization refers to

the proportion of the total volume occupied by boxes on the pallet relative to the total available space on the pallet. Besides, We define a stable state on the pallet as follows: when the agent places a box, we record the spatial position and orientation of the box at the moment of placement. At each timestep, after placing a new box, we check the difference between the current position and orientation of all boxes on the pallet and their recorded placement positions. If the deviation for any box exceeds a certain threshold, the state is considered unstable.

State. The state $s_t = \{C_t, d_t\}$ is composed of two parts: the first part is the pallet configuration C_t, and the second part is the properties of the boxes in the buffer d_t.

We represent the pallet's state using voxels, since we need to account for the intrinsic properties of the boxes. More specifically, in addition to the dimensions of the boxes, we also consider their density and rigidity. Therefore, we quantify the pallet configuration with a 2-channel 3D array, stands for the density distribution and rigidity distribution. d_t is a 1D array composed of information from N boxes, where each box's properties include length, width, height, density, and rigidity. Therefore, it is an array of size $5N$, where N represents the buffer capacity.

Action. It should include the following information: which box from the buffer to select, how to rotate the box, and where to place it on the pallet. We assume that after rotation, each box remains parallel to the boundary faces of the pallet, resulting in six possible orientations for each box. Additionally, we consider that new boxes cannot be placed unsupported in mid-air, so the placement is determined by the (x, y) coordinates. We discretize the pallet's length and width into l_p and w_p, respectively, giving $l_p \times w_p$ possible placement positions for each box.

As a result, the action a belongs to $\mathbb{R}^{N\times 6\times l_p\times w_p}$. For instance, with $N = 5$ and $l_p = w_p = 25$ [6], this results in 18,750 possible outcomes for the action. The vast size of this action space greatly increases the complexity of the RL problem, making the optimization process substantially more challenging.

B. Reward Design

We encourage the planner to improve space utilization, as long as it is stable. Therefore, we use the proportion of the box's volume placed at each step relative to the total space as the reward for each step. The reward function we designed is as follows:

$$r(s_t) = \mathbb{1}(s_t) \cdot \frac{V_{box}}{V_{max}} \tag{1}$$

V_{box} refers to the volume of the box selected at timestep s_t, while V_{max} represents the theoretical maximum volume that the pallet can accommodate. $\mathbb{1}(s_t)$ serves as an indicator function that validates the physical stability of the boxes on the pallet at state s_t.

In addition, to improve efficiency, we introduced a penalty term. When the number of boxes in the buffer is less than N, if the selected index does not correspond to any box, then $r(s_t) = -1$.

Another factor that affects the overall reward is the episode length. There are three reasons for an episode to end: after selecting a box, the action masking model determines that no feasible placement points exist; placing the box results in instability; or all the boxes have been placed on the pallet.

C. Integrating the Action Masking Model into the RL training process

At each timestep t, after the policy outputs an action, the corresponding box is selected from the buffer based on the action's information. The box's properties are then combined with the current pallet configuration to form a multi-channel 3D array, which is fed into the action masking model. The action masking model outputs an action space mask, indicating which points are considered stable and which are unstable at that moment. We refer to the set of stable points as the 'feasible set'. The model finds the point closest to the action's chosen position within the feasible set (if there are multiple nearest points, one is randomly selected). The box is then placed in the environment at this selected point within the feasible set.

D. Online action masking model learning

We choose to use a neural network as the action masking model and train it using an online approach because heuristic methods have numerous limitations that render them inadequate for our task setting.

Heuristic methods inherently face many limitations when addressing normal online 3D BPP. For example, the necessity for multiple hyperparameters in most action masking methods poses a challenge in tuning these parameters to accommodate diverse scenarios, and they fail to account for uncertainties inherent in real-world palletization execution [6]. When considering the density and rigidity of the boxes, the heuristic methods used in previous works become unusable. In the original scenario, the feasible set identified by the heuristic action masking model was a subset of the true feasible set. However, in this new scenario, they become an intersection, making the heuristic action masking model highly inefficient. This inefficiency is evident in the experimental results presented in Section IV. Designing a new heuristic method that incorporates both density and rigidity manually is cumbersome and lacks generalizability.

Therefore, we need a new paradigm to solve this problem, one that relies on directly acquiring experience from the simulator. We propose the online action space mask learning paradigm, which incorporates supervised learning for the action masking model into the RL training loop, shown in Figure 1. Initially, both the policy network and the action masking model are randomly initialized. During the RL policy rollout, we record data for action masking model training. Those data contain two parts: the configurations of the pallet C_t and the properties of the next to be placed box b_t. After a certain number of timesteps, we update the action masking model along with the policy network. The update process for the action masking model involves two steps: first, generating an annotation g_t for each recorded

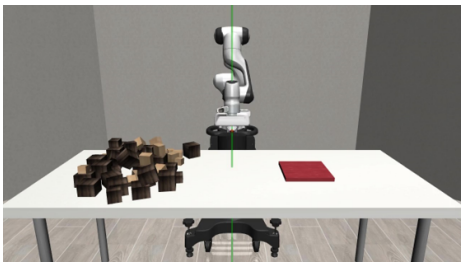

Fig. 2: Visualization of our simulated palletization environment in MuJoCo [23]. Although 40 boxes are displayed for illustrative purposes, the robot is programmed to perceive and interact with only N boxes within the buffer area. The arrangement of the boxes is randomized and unknown, shuffled anew for each RL episode.

data point (C_t, b_t) from the previous rollout phase; second, appending these data points along with their annotations to the dataset, called feasible dataset, and using this feasible dataset to train the action masking model.

We use multiple parallel simulators specifically created to generate annotations. In these simulators, the scene is first reconstructed based on the assigned (C_t, b_t), after which the box is tested at every possible position. It is important to note that some positions can be judged using simple heuristics without requiring the simulator, such as a box placed directly on the pallet surface being inherently stable, or a box with less than 25% support being inherently unstable.

There are some important details regarding data selection and management. For episodes that end due to instability, the corresponding data is always recorded. For regular rollout data, there is a 0.1 probability of being recorded. This ensures that the action masking model focuses on learning cases where its predictions were less accurate, while still capturing regular data to avoid overfitting. For managing the feasible dataset, we use a deque data structure, meaning that data is managed in a first-in-first-out manner. This allows the action masking model to encounter more diverse data, helping to address the issue of out-of-distribution data.

We demonstrate the effectiveness of our proposed method in the following section.

IV. EXPERIMENTAL VALIDATIONS

In this section, we present a thorough quantitative analysis of our proposed approach, and formulate our experiments to answer the following questions:

1) Does the action space mask training method mentioned earlier eventually converge and learn a relatively accurate result?
2) How are the changes in the performance of the policy and the action masking model related?
3) Does the online action masking model learning method enhance the RL training process, resulting in a superior policy?

A. Simulation Experiments

Setup. To explore the task planning challenges associated with the palletization problem, we developed a simulated palletization task in MuJoCo [23] [24], reflecting a realistic logistic scenario, as depicted in Figure 2. We adjusted the parameters in the simulation environment that solve for the box deformation under contact forces to give the boxes different levels of rigidity. There are four different types of boxes in the simulation environment: two soft, low-density boxes and two hard, high-density boxes. The size, density, rigidity, and quantity of boxes in the simulation are shown in Table I.

TABLE I: Box Specifications in the Simulation Environment.

Dimension (inches)	Density	Rigidity	Counts
$6 \times 6 \times 4$	500	0.5	10
$6 \times 6 \times 6$	500	0.5	10
$6 \times 6 \times 6$	5000	3	10
$8 \times 6 \times 6$	5000	3	10

If a hard box is placed on top of a soft box in the simulation environment, the latter will collapse, which is considered an unstable situation. The pallet is specified to have dimensions of 25×25 inches, with a stipulation that the maximum height of stacked boxes cannot exceed 20 inches. For our experiments, we applied a discretization resolution of 1 inch to both the boxes and the pallet. In all the subsequent simulation experiments mentioned, we have chosen a buffer size of $N = 5$. And we use PPO [25] as the RL algorithm for all experiments.

At each step of the planning process, the task planner receives the current pallet configuration, represented as a density distribution, along with the properties of the next N boxes in the buffer, which include their dimensions and density. The planner's task is to select one of the N boxes, determine its orientation, and place it on the pallet. Since our focus is on planning rather than manipulation, the actual 'pick-rotate-place' actions by the robot are omitted. Instead, the selected box is immediately positioned at the goal pose determined by the planner, to accelerate the learning process.

To better simulate real-world palletization, we added noise to the box placement when generating feasible annotations: positional noise on the xy plane, $\delta_t \sim N(0, 0.05)$, measured in inches, and rotational noise around the z-axis, $\delta_r \sim N(0, 5)$, measured in degrees.

Action space mask training results. During the training of the policy, we used a deque with a size of 16,000 as a feasible dataset for training the action masking model.

For each policy rollout step, if the step is unstable and causes the episode to end, that data will definitely be recorded in the dataset. If it is just a normal step, there is a 10% chance it will be recorded in the dataset. In this way, we can make the action masking model focus more on cases where it failed to make correct judgments, ultimately leading to more accurate performance.

We split the dataset into an 8:2 ratio for the train set and validation set, using the model's IoU score on the validation set to evaluate its performance. At each policy update, we simultaneously update the action masking model. The IoU score is very low at the beginning. As the total number of training epochs increases and the model encounters more

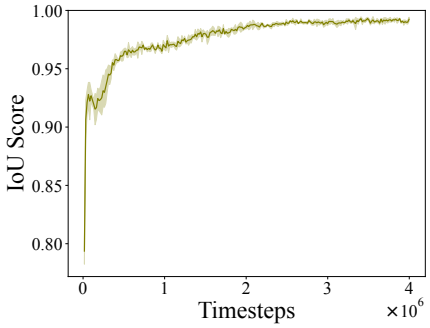

Fig. 3: IoU score on validation set. After rolling out for a certain number of timesteps, we update the action masking model and calculate the IoU score on the validation set. Results are averaged over 5 random seeds.

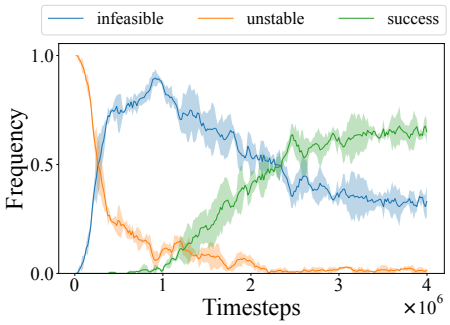

Fig. 4: The frequency change curve for the three types of episode end reasons. During training, we record the frequency of each episode end reason, which is represented by the curve shown. Results are averaged over 5 random seeds.

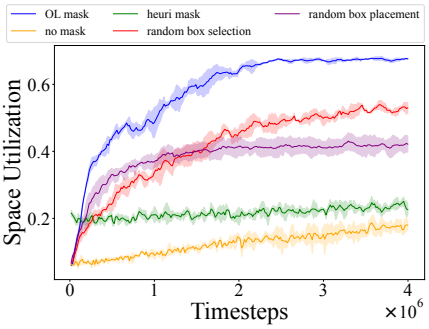

Fig. 5: Space utilization. Our method (OL Mask) demonstrates faster convergence and achieves better space utilization compared to the other methods. Results are averaged over 5 random seeds.

TABLE II: Performance evaluation of different planners under various methods.

	Nomask	Random Selection	Random Placement	Heuristic mask	**OL Mask (Ours)**
Space utilization	0.18	0.53	0.41	0.26	**0.68**
Infeasible Rate	**0.00**	0.84	0.95	0.05	0.34
Unstable Rate	1.00	0.14	0.05	0.95	**0.01**
Mean Episode Length	12.3	31.8	25.8	15.9	**39.4**
Mean Episode Reward	0.18	0.49	0.41	0.26	**0.66**
Success Rate	0.00	0.02	0.00	0.00	**0.65**

new data, the capability of the action masking model gradually improves. The action masking model using the online learning method began to converge after approximately 2 million rollout timesteps, as shown in Figure 3. Eventually, the IoU score converges to around 98% on the validation set

Such a high IoU score indicates that the action masking model is already capable of determining which positions are stable or unstable based on the current situation of boxes on the pallet and the information of the next box.

Guidance from the action space mask. To study the relative performance of the action masking model and the policy, we recorded the probability of each of the three end reasons occurring in each episode during training, as shown in Figure 4. Each episode can be ended for the following reasons.

- *Infeasible:* The action masking model indicates that there is no feasible point on the pallet;
- *Unstable:* After placing each box, the situation on the pallet becomes unstable, resulting in a collapse.
- *Success:* All 40 boxes have been placed properly.

Initially, because the action masking model is randomly initialized, its judgments on stability are very inaccurate. As a result, in the beginning, every episode ends as unstable. However, as the action masking model continues to be trained, its accuracy gradually improves. At this point, since the policy has not yet learned how to place boxes densely, the frequency of cases where boxes are selected but have no place to be placed increases significantly. Finally, as the policy, guided by the accurate action space mask, continuously attempts different placements, it learns a compact and stable way to place boxes, leading to a significant increase in the success rate. Since the training of the action masking model is supervised learning, it is more likely to achieve better performance more quickly. Therefore, it converges first and then guides the policy to achieve good performance.

Ablative experiments. To verify the importance of both RL planning and online action space mask learning in our method, we selected four baselines.

- Nomask: The action masking is not applied during the RL training process.
- Heuristic mask: We use the heuristic mask from [4], which considers two criteria: the support ratio and the number of support corners. A placement is considered stable in any of the following three cases: support ratio $> 0.6 \wedge$ support corners $= 4$; support ratio $> 0.8 \wedge$ support corners ≥ 3; support ratio > 0.95
- Random box selection: In this method, an online training action space mask is used, but the selection of boxes is independent of the actions output by the RL algorithm and is instead randomly drawn from the buffer.
- Random box placement: Under the premise of using an online training action space mask, the selection of boxes is determined by the policy, but their placement is random.

We use the mean episode space utilization to measure the overall quality of the final policy, as shown in Figure 5. Initially, the space utilization of the three methods using online action masking model learning ('OL mask', 'random box selection', 'random box placement') is the same as

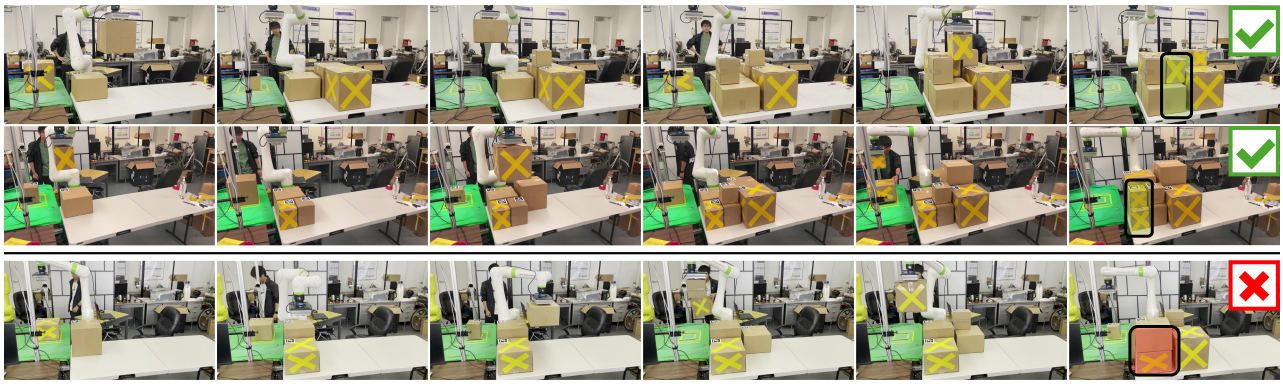

Fig. 6: Real-world experiments with each row as a planning sequence. The boxes with yellow cross tape are considered to have low density and be soft, while other boxes are considered to have high density and be hard. 1) The first two rows show the planner trained using our method, successfully placing 9 boxes on the pallet while maintaining stability, under two different box arrival sequences. 2) The bottom row shows the planner trained with a heuristic mask, which, after stacking 5 boxes, incorrectly placed a high-density, high-rigidity box on top of a low-density, low-rigidity box, which is considered a failure case of unstable. The unstable position is marked with a red cross.

that of the no mask method, due to their action masking models being randomly initialized. At the beginning, the heuristic mask method achieves the highest space utilization. However, as training progresses, the space utilization of the three online action masking methods quickly surpasses that of the other two methods. Among all the methods, the space utilization for the 'nomask' method remains the lowest throughout the training process, while the space utilization for the 'nomask' method remains the highest. After 4M timesteps, the three methods using online learning action space mask perform significantly better than the other two methods. This demonstrates the importance of the action space mask and shows that an accurate action space mask can accelerate convergence and greatly improve the planner's capability. Comparing our method with the 'random box' and 'random place' methods also highlights the importance of RL in the overall approach, demonstrating that each component of our method is indispensable as part of the whole.

B. Real World Experiments

We further implement the policy and action masking model learned in simulation onto a real robot arm. Since the suction-type gripper we used is relatively large, the minimum size of the boxes is also larger. Due to the limited range of motion of the robotic arm, it is not possible to stack 40 boxes as in the simulation when dealing with larger boxes. Therefore, we retrain the policy and action masking model to accommodate the real-world setting.

Since we are focused solely on the planning aspect of palletization, we assume that the perception task should be handled earlier in the pipeline. We use AprilTags [26] to record the size, density, and softness of each box. Near the buffer, we installed an Intel RealSense camera, which scans the AprilTags and compares the data with pre-stored information in the computer to obtain the details of the boxes currently in the buffer. Due to spatial constraints, we set the buffer size to 1. Additionally, since the robotic arm can easily rotate boxes around the z-axis during operation, we only consider rotation around the z-axis.

The pallet we considered has both a length and width of 24 inches, with a maximum allowable stacking height of 20 inches. We consider a total of 9 boxes, but there are 8 different types of boxes. Finally, our planner successfully places all 9 boxes on the pallet and achieves a stable solution, with a space utilization rate of 70%. The compact and stable configuration highlights the effectiveness and reliability of the learned task planner, shown in Figure 7.

Fig. 7: Final pallet configuration viewed from 4 perspectives. In a stable situation, boxes with high density and rigidity should be placed under heavy and soft ones.

V. Conclusion and Future Work

In this paper, we introduce a variant of the online 3D bin packing problem that incorporates intrinsic box properties beyond dimensions, to more closely resemble real-world scenarios. To address this new challenge, we propose an online action masking model learning method, which completely eliminates the need for heuristic methods, offering a more generalized approach. Simulations validate the effectiveness of our method, showing superior performance compared to other approaches. In the future, we may explore even more realistic scenarios, such as handling friction between boxes and assuming the pallet may experience sudden shaking, which simulates the acceleration and deceleration phases encountered during real-world transportation.

REFERENCES

[1] E. Lamon, M. Leonori, W. Kim, and A. Ajoudani, "Towards an intelligent collaborative robotic system for mixed case palletizing," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9128–9134.

[2] R. Szczepanski, K. Erwinski, M. Tejer, A. Bereit, and T. Tarczewski, "Optimal scheduling for palletizing task using robotic arm and artificial bee colony algorithm," *Engineering Applications of Artificial Intelligence*, vol. 113, p. 104976, 2022.

[3] F. Wang and K. Hauser, "Robot packing with known items and nondeterministic arrival order," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 4, pp. 1901–1915, 2020.

[4] H. Zhao, Q. She, C. Zhu, Y. Yang, and K. Xu, "Online 3d bin packing with constrained deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, 2021, pp. 741–749.

[5] H. Zhao, Y. Yu, and K. Xu, "Learning efficient online 3d bin packing on packing configuration trees," in *International conference on learning representations*, 2021.

[6] Z. Wu, Y. Li, W. Zhan, C. Liu, Y.-H. Liu, and M. Tomizuka, "Efficient reinforcement learning of task planners for robotic palletization through iterative action masking learning," *arXiv preprint arXiv:2404.04772*, 2024.

[7] C. T. Ha, T. T. Nguyen, L. T. Bui, and R. Wang, "An online packing heuristic for the three-dimensional container loading problem in dynamic environments and the physical internet," in *Applications of Evolutionary Computation: 20th European Conference, EvoApplications 2017, Amsterdam, The Netherlands, April 19-21, 2017, Proceedings, Part II 20*. Springer, 2017, pp. 140–155.

[8] F. Wang and K. Hauser, "Stable bin packing of non-convex 3d objects with a robot manipulator," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8698–8704.

[9] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[10] G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, T. Mann, T. Weber, T. Degris, and B. Coppin, "Deep reinforcement learning in large discrete action spaces," *arXiv preprint arXiv:1512.07679*, 2015.

[11] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser, *et al.*, "Starcraft ii: A new challenge for reinforcement learning," *arXiv preprint arXiv:1708.04782*, 2017.

[12] D. Ye, Z. Liu, M. Sun, B. Shi, P. Zhao, H. Wu, H. Yu, S. Yang, X. Wu, Q. Guo, *et al.*, "Mastering complex control in moba games with deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 6672–6679.

[13] H. Zhao, C. Zhu, X. Xu, H. Huang, and K. Xu, "Learning practically feasible policies for online 3d bin packing," *Science China Information Sciences*, vol. 65, no. 1, p. 112105, 2022.

[14] O. Faroe, D. Pisinger, and M. Zachariasen, "Guided local search for the three-dimensional bin-packing problem," *Informs journal on computing*, vol. 15, no. 3, pp. 267–283, 2003.

[15] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[16] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[17] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*. Springer, 2016, pp. 424–432.

[18] T. G. Crainic, G. Perboli, and R. Tadei, "Ts2pack: A two-level tabu search for the three-dimensional bin packing problem," *European Journal of Operational Research*, vol. 195, no. 3, pp. 744–760, 2009.

[19] K. Kang, I. Moon, and H. Wang, "A hybrid genetic algorithm with a new packing strategy for the three-dimensional bin packing problem," *Applied Mathematics and Computation*, vol. 219, no. 3, pp. 1287–1299, 2012.

[20] R. Hu, J. Xu, B. Chen, M. Gong, H. Zhang, and H. Huang, "Tap-net: transport-and-pack using reinforcement learning," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–15, 2020.

[21] J. Zhang, B. Zi, and X. Ge, "Attend2pack: Bin packing through deep reinforcement learning with attention," *arXiv preprint arXiv:2107.04333*, 2021.

[22] K. Karabulut and M. M. İnceoğlu, "A hybrid genetic algorithm for packing in 3d with deepest bottom left with fill method," in *International Conference on Advances in Information Systems*. Springer, 2004, pp. 441–450.

[23] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 5026–5033.

[24] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu, "robosuite: A modular simulation framework and benchmark for robot learning," in *arXiv preprint arXiv:2009.12293*, 2020.

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[26] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4193–4198.