GOPT: Generalizable Online 3D Bin Packing via Transformer-Based Deep Reinforcement Learning

Heng Xiong, Changrong Guo, Jian Peng, Kai Ding, Wenjie Chen, Xuchong Qiu, Long Bai, and Jianfeng Xu

***Abstract*—Robotic object packing has broad practical applications in the logistics and automation industry, often formulated by researchers as the online 3D Bin Packing Problem (3D-BPP). However, existing DRL-based methods primarily focus on enhancing performance in limited packing environments while neglecting the ability to generalize across multiple environments characterized by different bin dimensions. To this end, we propose GOPT, a generalizable online 3D Bin Packing approach via Transformer-based deep reinforcement learning (DRL). First, we design a Placement Generator module to yield finite sub-spaces as placement candidates and the representation of the bin. Second, we propose a Packing Transformer, which fuses the features of the items and bin, to identify the spatial correlation between the item to be packed and available sub-spaces within the bin. Coupling these two components enables GOPT's ability to perform inference on bins of varying dimensions. We conduct extensive experiments and demonstrate that GOPT not only achieves superior performance against the baselines, but also exhibits excellent generalization capabilities. Furthermore, the deployment with a robot showcases the practical applicability of our method in the real world.**

***Index Terms*—Manipulation planning, reinforcement learning, robotic packing.**

I. Introduction

With the prosperity of the global trade and e-commerce market, warehouse automation has developed rapidly in recent years. Efficient object placement in the warehouse through optimal packing strategies can bring numerous benefits, such as reduced labor requirements and cost savings [1].

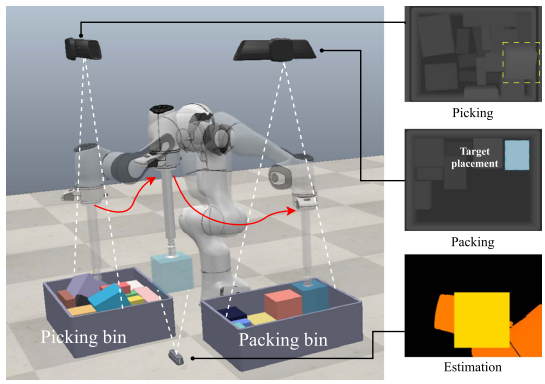


Fig. 1. Robot picking and packing pipeline. Left: A robot randomly picks an item from a cluttered collection of boxes and packs it in a compact manner, and three RGB-D cameras are mounted. Right: Two overhead cameras observe the status of the two bins, respectively, and one up-looking camera estimates the dimension of the picked item.

Fig. 1 illustrates an example of item picking and packing using a robotic arm. In this letter, it is assumed that the robot picking is well implemented. Researchers have commonly addressed the placement challenge in robot packing by formulating it as an online 3D Bin Packing Problem (3D-BPP) [2], [3]. As one of the classic combinatorial optimization problems, 3D-BPP strives to place a set of known cuboid items in an axis-aligned fashion into a bin to maximize space utilization. However, observing all items and obtaining full knowledge about them is challenging in many real-world scenarios. The online 3D-BPP is a more practical variant of 3D-BPP that refers to packing items one by one under the observation of only the incoming item.

Due to the limited knowledge, the online 3D-BPP cannot be solved by exact algorithms [4]. Researchers have previously focused on developing heuristics with the greedy objectives for the problem, which are designed by abstracting the experience of human packers [5]. However, while intuitive, these heuristics typically yield sub-optimal solutions. In recent years, there has been an emerging research interest in resolving online 3D-BPP via deep reinforcement learning (DRL) [2], [3], [6], [7], and indeed, DRL-based methods demonstrate impressive performance. Nevertheless, it is noteworthy that the training process often encounters challenges in reaching convergence [2], [8], and these methods struggle to generalize effectively across

Received 22 June 2024; accepted 7 September 2024. Date of publication 25 September 2024; date of current version 10 October 2024. This article was recommended for publication by Associate Editor Heping Chen and Editor Chao-Bo Yan upon evaluation of the reviewers' comments. This work was supported by the National Key R&D Program of China under Grant 2022YFB4700300. *(Corresponding author: Jianfeng Xu.)*

Heng Xiong, Changrong Guo, Jian Peng, and Long Bai are with the State Key Laboratory of Intelligent Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: xiongheng@hust.edu.cn; guochangrong@hust.edu.cn; peng_jian@hust.edu.cn; bailong@hust.edu.cn).

Kai Ding and Xuchong Qiu are with BOSCH Corporate Research, Shanghai 200335, China (e-mail: kai.ding@cn.bosch.com; xuchong.qiu@cn.bosch.com).

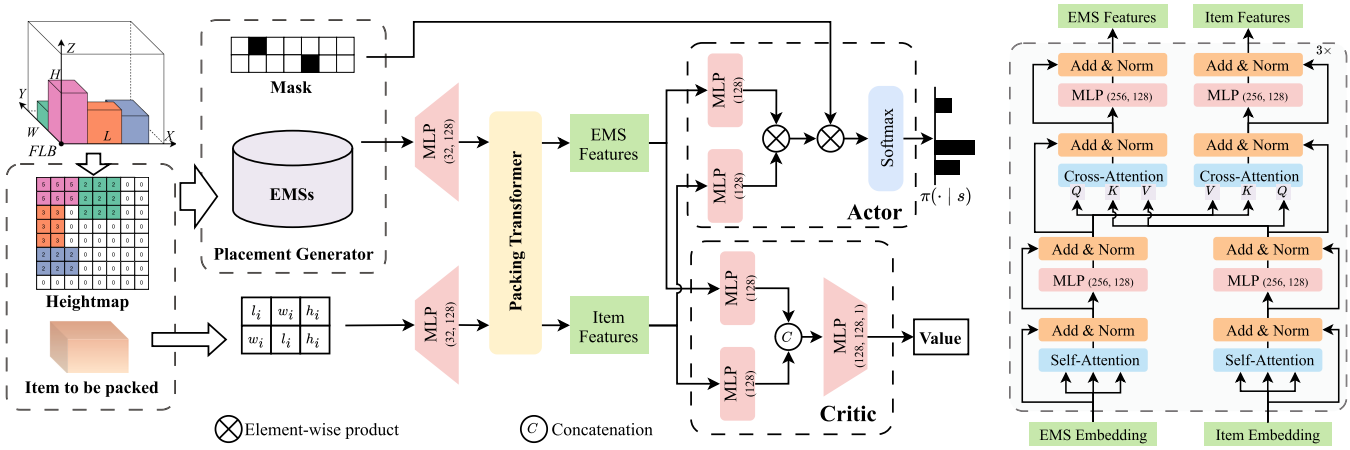
Wenjie Chen is with the State Key Laboratory of High-end Heavy-load Robots, Midea Group, Foshan 528300, China, and also with Midea Corporate Research Center, Foshan 528311, China (e-mail: chenwj42@midea.com).

Jianfeng Xu is with the State Key Laboratory of Intelligent Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China, and also with HUST-Wuxi Research Institute, Wuxi 214174, China (e-mail: jfxu@hust.edu.cn).

The source code will be publicly available at https://github.com/Xiong5Heng/GOPT.

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3468161, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3468161



(a) The framework of GOPT (b) Packing Transformer

Fig. 2. Overview of our method. (a) In the GOPT, the inputs comprise the item to be packed and the current heightmap of the bin, wherein each cell's value represents the respective height. Utilizing the placement generator, a set of EMSs is produced, along with a pairwise action mask between each EMS and the optional orientation of the item. After that, we separately encode the EMSs and the item and then fuse the features using the packing transformer, of which outputs are fed into the actor and critic networks to generate logits of all actions and estimate the state-value function; (b) Depicts the details of the proposed packing transformer. The transformer comprises three stacked blocks, each containing two self-attention and two cross-attention layers.

diverse packing scenarios, especially those characterized by different bin dimensions. These limitations substantially curtail the broader applicability of DRL in typical use cases. More specifically, the current state-of-the-art DRL-based methods can only perform inference on bins of the same size as those they are trained on [3], [9]. Trained models are not transferable to bins of different sizes. Additionally, the inherent dependence of the packing action space size on bin dimensions in these methods presents significant challenges for model convergence, especially when dealing with larger bins [10].

Motivated by the aforementioned limitations, this letter proposes GOPT, a generalizable online 3D Bin Packing approach via Transformer-based DRL, as shown in Fig. 2. In GOPT, a *Placement Generator (PG)* module first adopts a heuristic to generate a fixed-length set of free sub-spaces within the current bin as placement candidates, which ensures controllability over the size of the packing action space. Both the placement candidates and the item to be packed are collectively defined as the state of the Markov Decision Process (MDP). Then, GOPT incorporates a novel packing policy network that integrates a *Packing Transformer* module. This module enhances GOPT's generalizability by intrinsically identifying the spatial correlation between the current item and the available sub-spaces, as well as the relations among these sub-spaces, which are derived from the PG module. The Packing Transformer employs self-attention layers and bi-directional cross-attention layers to extract features as inputs to the reinforcement learning policy.

Experiments show that our method outperforms the state-of-the-art packing methods in terms of space utilization and the number of packed objects. To the best of our knowledge, our work is the first to provide the generalization capability to infer across various bins with a trained model while maintaining high performance. We also deploy our packing planning method in a robotic manipulator to demonstrate its practical applicability in the real world.

In summary, our main contributions are: (1) GOPT, a novel method for online 3D-BPP that enlarges the packing performance and generalization; (2) A Placement Generator module to modulate the packing action space and represent the state of the bin; (3) A network called Packing Transformer, which captures the relations between the current item and the available sub-spaces, as well as interrelations among sub-spaces; (4) Extensive experimental evaluations comparing GOPT with baselines.

II. Related Work

The 3D-BPP is a classical optimization problem and is known to be strongly NP-hard [11]. We herein briefly review related heuristic and DRL-based methods.

A. Heuristic Methods

Early works primarily focus on designing efficient heuristics for their simplicity. Researchers attempt to define some packing rules distilled from human workers' experience, such as First Fit [12], Best Fit [13], and Deepest-Bottom-Left-Fill [14]. Corner points (CP) [15], extreme points (EP) [16], empty maximal spaces (EMS) [17], and internal corners point (ICP) [18] endeavor to represent potential free spaces where items can be packed for enhancing heuristic methods. For instance, Ha et al. [5] propose OnlineBPH, which selects one EMS to minimize the margin between the faces of the item to be packed and the faces of the EMS. Yarimcam et al. [19] provide heuristics expressed in terms of policy matrices by employing the Irace parameter tuning algorithm [20]. Wang et al. [21] propose Heightmap-Minimization (HM) which favors the placement that minimizes occupied volume. To mitigate the uncertainties originating from the real world, Shuai et al. keep deformed boxes stacked close to enhance stability [22]. Hu et al. develop

a Maximize-Accessible-Convex-Space (MACS) strategy to optimize the available empty space for packing potential large future items [23]. These methods are intuitive and effective; however, they rely on hand-crafted rules and lack the capacity to demonstrate superior performance consistently across diverse problem settings. Our work draws on the representation of empty spaces in heuristics, but uses DRL to learn packing patterns without being limited by domain expert knowledge.

B. DRL-Based Methods

DRL has shown promise in solving certain combinatorial optimization problems [24], [25]. Therefore, there is a trend to use DRL to solve the 3D-BPP recently. Que et al. [26] tackle the offline 3D-BPP with variable height by using DRL with Transformer structure to sequentially address subtasks of position, item selection, and orientation. Instead, we focus on the online 3D-BPP and determine the position and orientation simultaneously. To the best of our knowledge, Deep-Pack [27] is the first to use a DRL-based model to solve a 2D online packing problem, with potential extensions to the online 3D-BPP. It takes an image showing the current state of the bin as input and outputs the pixel location for packing the incoming item. Verma et al. [6] combine a search heuristic with DRL and propose a two-step strategy for solving the problem with any number and size of bins. Zhao et al. [2], [10] formulate the problem as a constrained MDP and adopt ACKTR method [28] to train a CNN-based DRL agent. In [2], the DRL agent comprises an actor, a critic, and a predictor to estimate action probabilities, value, and feasibility mask, respectively. It is then improved by decomposing the packing action into the length and width dimensions and orientation to reduce action space [10]. They subsequently introduce the Packing Configuration Tree (PCT) based on heuristic search rules and incorporate it into a DRL agent [8]. The agent employs Graph Attention Networks [29] as the policy and is also trained with ACKTR. To investigate the synergies of heuristics and DRL, Yang et al. [7] propose PackerBot, which utilizes heuristic reward to assist the DRL agent to perform better. Xiong et al. [3] introduce a candidate map mechanism to reduce the complexity of exploration and improve performance for the CNN-based DRL agent trained with A2C [30]. These methods usually concatenate features of the item and the bin directly to learn policies. In contrast, GOPT first proposes free sub-spaces within a bin and utilizes a modified Transformer to discern the relations among these spaces and the relations between them and the current item. Our method ensures generalizability across diverse packing environments.

III. Methodology

A. Problem Description

As shown in Fig. 1, a robot randomly picks an object from an unstructured pile with a set of box-shaped items of various dimensions. The complete knowledge about all items is unavailable in advance. One camera measures the dimensions of the picked item, which is then placed into the packing bin. This specific scenario can be characterized as an online 3D-BPP. The objective is to place as many items into the bin as possible and maximize the bin's space utilization.

We define the front-left-bottom (FLB) vertex of the bin with dimensions (L, W, H) as the origin $(0, 0, 0)$, and the directions along the length, width, and height as X, Y, and Z directions, respectively, as shown in Fig. 2(a). For items, (x_t, y_t, z_t) denotes the FLB coordinate of the t-th item with dimensions (l_t, w_t, h_t).

In the robot packing task, the following physical constraints must be taken into consideration.

Orthogonal placement: Items are placed orthogonally into the bin, and their sides are aligned with the bin's sides.

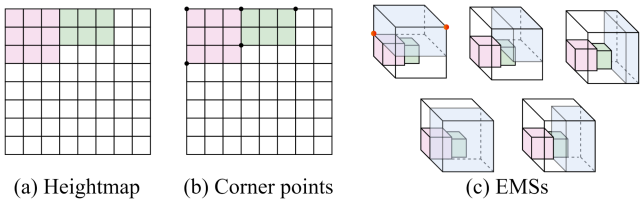
Optional orientation: Items are placed in an upright manner; in conjunction with the first constraint, items have just two distinct vertical in-plane orientations, either 0° or 90°.

Static stability: During the process of packing, items must remain stable under gravity and inter-item forces. For computational efficiency, an item is considered stable if the projection of its geometric center onto its bottom falls inside the support polygon which is formed by the convex hull of all horizontal support points of this item [23].

B. Placement Generator

For the selected item to be packed, we predict the horizontal position (x_t, y_t) and the corresponding orientation of its placement in the bin. The vertical position z_t is analytically determined by the lowest placement position due to gravity. As aforementioned, there are two possible orientations for one item. Therefore, when placing an item into a bin with dimensions (L, W, H), it results in a total number of $L \times W \times 2$ possible placements [2]. On the one hand, this quantity is unbearable for the packing problem with the sequential-decision nature because it will grow exponentially with larger bin dimensions. On the other hand, some are inevitably unproductive for the item to be packed within this placement set.

With the aim of constraining the potentially large placement search space, we design a Placement Generator (PG) module to produce a finite and efficient placement subset based on the incoming item and current bin configuration. We first explicitly represent the real-time status of the bin by utilizing the heightmap. Other methods that leverage planned placements for previous items as the representation [8] lack feedback and closed-loop control. In contrast, the heightmap can be derived from the visual observation captured by a camera conveniently when deploying PG in a real-world robot packing task. Drawing from the empty maximal space (EMS) scheme for managing the empty spaces in a bin [17], [31], candidate placements are computed based on the current state. Specifically, we identify corner points by detecting height variation along the heightmap's X and Y directions. EMSs are then generated by extending unit rectangles from each corner and halting when encountering higher elevation (Fig. 3). Each EMS can be defined by its FLB vertex and the corresponding opposite vertex as depicted in Fig. 3(c). The resulting 6-dimensional vector is normalized to $[0, 1]$, regardless of the dimensions of the bin. We obtain an EMS subset with controllable size and rank them by height value, denoted as $\{E_i\}_{i=1}^{N}$. Finally, given an item to be packed, we



(a) Heightmap (b) Corner points (c) EMSs

Fig. 3. Illustration of the EMS generation procedure. (a) In an example scene with two placed items, the heightmap indicates the current height of stacked items in each grid cell; (b) Five corner points (black dots) are detected at this heightmap; (c) Based on these points, the corresponding largest inscribed rectangles (blue) within the bin are generated, namely EMSs. Taking the first EMS as an example, it is defined by two red vertices of the blue rectangles.

check the feasibility of each EMS following Section III-A and produce a pairwise mask between each EMS and orientation. When packing the item within the bin, we select an appropriate EMS and orientation and align the item's and the EMS's FLB vertices.

C. Reinforcement Learning Formulation

DRL problems are commonly modeled as a Markov Decision Process (MDP). An MDP with parameters $\langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$ is utilized to characterize the packing environment in this letter, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, +\infty)$ stands for the transition probabilities, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the scalar reward function, and $\gamma \in (0, 1]$ is the discount factor for balancing the near-term and long-term rewards in DRL. Reinforcement learning algorithms aim to learn a policy $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which determines the probability of selecting an action a given a state s. The objective of the policy is to maximize the cumulative discounted reward over an episode, expressed as $\sum_t \gamma^t r_t$, where t denotes the time step, and r_t, a_t, and s_t represent the reward, action, and state at time step t, respectively. In the following, we formulate the online 3D-BPP as an MDP for DRL training.

State: At each time step t, the policy receives a state s_t, comprising the incoming item to be packed $s_{t,item}$ and the current bin configuration $s_{t,bin}$. For the first part, the dimension of the item (l_t, w_t, h_t) is essential. Some studies [3], [7] employ this three-dimensional vector explicitly as the item representation, while others prefer a three-channel map for the convenience of neural network design [2], [9]. In the map representation, each channel is assigned l_t, w_t, and h_t, respectively. To account for both the geometry and optional orientations, we propose an item representation which is a 2×3 matrix, $s_{t,item} = \begin{bmatrix} l_t & w_t & h_t \\ w_t & l_t & h_t \end{bmatrix}$, where (l_t, w_t, h_t) and (w_t, l_t, h_t) represent the dimensions of the item after rotating it by 0° and 90°. For the second part, the existing methods include the heightmap [3], the list of packed items [8], and the weighted 3D voxel grid [9]. We choose to leverage the proposed PG (Section III-B) to produce a sequence of EMSs satisfying placement constraints as the bin's configuration. The sequence is padded or clipped to a fixed length N with dummy EMSs, i.e. $s_{t,bin} = \{E_i\}_{i=1}^{N}$.

Action: Given the packing state $s_t = (s_{t,item}, s_{t,bin})$, the action a_t involves selecting both an orientation and an EMS for the current item from the sequence of available EMSs. The size of the action space $\mathcal{A}$ depends solely on the length of the sequence and the number of optional orientations, i.e., $\|\mathcal{A}\| = 2N$, irrespective of the bin dimensions. During training, we select the action a_t according to the probability distribution over actions $\pi(\cdot \mid s_t)$, where $\cdot$ represents the set of all possible placements in s_t. During testing, we select the action in a deterministic manner by choosing the placement with maximum probability in $\pi(\cdot \mid s_t)$. Note that the probability distribution applying the pairwise action mask between EMSs and orientations ensures that the policy samples valid actions unless no EMS satisfies the constraints.

State-Transition: In our setting, the transition model is assumed to be deterministic, implying that a specific pair (s_t, a_t) consistently leads to the same subsequent state s_{t+1}.

Reward: The target of the packing problem is to maximize the space ratio of the bin. Hence, we formulate the reward as the step-wise enhancement in space utilization, represented as $r_t = \frac{l_t \cdot w_t \cdot h_t}{L \cdot W \cdot H}$. This dense reward encourages the DRL agent to perform more steps in an episode, thereby leading to more packed items and greater space utilization.

D. Network Architecture

The design of a neural network architecture for the DRL agent is important because the chosen architecture affects the agent's learning and generalization capabilities across varied environments. A simplistic network would be to concatenate the bin and item representations [2] or embeddings [7]. However, this method results in a model whose convolutional and linear layer sizes are contingent upon the dimensions of the bin, rendering the trained model impractical for application across different bins.

To overcome the challenge of generalization, we propose an attention-based network architecture that focuses on the correlation between the item and the bin's partial spaces. As illustrated in Fig. 2(a), this architecture comprises three primary components: the Packing Transformer, the actor network, and the critic network. Our network takes the bin representation $s_{t,bin} \in \mathbb{R}^{N \times 6}$ (i.e., a sequence of EMSs from PG) and the item representation $s_{t,item} \in \mathbb{R}^{2 \times 3}$ (i.e., item's dimensions) as inputs. These inputs are then individually processed by Multi-Layer Perceptrons (MLP), which are two-layer linear networks with LeakyReLU activation function. The embedding dimensions of both EMS and the item are set to 128. Subsequently, we then extract features from the embeddings using the designed Packing Transformer, inspired by cross-modality learning across language and vision [32]. The EMS and item features are then fed into the actor network to generate a probability distribution of potential actions, and fed into the critic network to estimate the expected cumulative reward based on the current state.

Packing Transformer is depicted in detail in Fig. 2(b). It is constructed by stacking multiple (three in practice) identical encoder blocks, each containing two self-attention layers, one bi-directional cross-attention layer, and four MLP blocks of two

layers comprising $\{128, 128\}$ neurons. The bi-directional cross-attention layer consists of two unidirectional cross-attention layers, one from EMS to item and the other from item to EMS. Residual connections and layer normalization (Norm) are applied after each layer. The self-attention layers play an important role in establishing the intrinsic connections between EMSs or item dimensions, while the bi-directional cross-attention layer facilitates the discovery of inner-relationships from one to another.

Actor and critic networks are both implemented with the MLP layers shown in Fig. 2(a). In the actor network, both the EMS and item features are processed through an MLP, and the results are multiplied to compute a score map of actions. This is followed by an element-wise multiplication with the action mask to eliminate infeasible actions.

E. Training Method

We employ the Proximal Policy Optimization (PPO) algorithm [33] to train the proposed GOPT. PPO is a popular on-policy reinforcement learning algorithm that alternates between collecting data via interactions with the environment and optimizing the following objective, which is approximately maximized in each iteration:

$$\mathcal{L}(\theta) = \hat{\mathbb{E}}_t[\mathcal{L}^{CLIP}(\theta) - c_1\mathcal{L}^{VF}(\theta) + c_2 S(\pi_\theta(\cdot \mid s_t))] \quad (1)$$

where θ represents the network parameters, c_1, c_2 are coefficients, $\mathcal{L}^{CLIP}(\theta)$ is the clipped surrogate objective, $\mathcal{L}^{VF}(\theta)$ is the squared-error loss for the value function, and S denotes the entropy of the policy. Specifically, the surrogate objective is defined as:

$$\mathcal{L}^{CLIP} = \hat{\mathbb{E}}_t[\min(p_t(\theta)\hat{A}_t, \text{clip}(p_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \quad (2)$$

where $p_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the action probability ratio between the current policy and the old policy, $\hat{A}_t$ is the estimation of the advantage function which we use Generalized Advantage Estimator (GAE) [34] method to compute, and ϵ indicates the clipped ratio which is used to limit the volume of update and stabilize learning procedure.

IV. Experiments

A. Implementation Details

Our method is implemented utilizing PyTorch and adopts the PPO algorithm within the Tianshou framework [35] for policy training. The maximum number of EMS is set to 80 during each packing step. We train the policy for 1000 epochs and collect a total of 40,000 environment steps over 128 parallel environments in every epoch. Policy updates occur after every 640 environment steps (calculated as 5×128 steps), with a batch size of 128. The Adam optimizer, coupled with a linearly descending learning rate scheduler starting from 7×10^{-5} is utilized for optimization. In terms of PPO loss calculation, the coefficients for value and entropy loss c_1, c_2 are 0.5 and 0.001, respectively, and the clipped ratio ϵ is 0.3. The discount factor γ is set to 1 to consider future and immediate rewards equally important. For policy updates, we use GAE with $\lambda_{GAE} = 0.96$. Our policy training is conducted on a computer equipped with an NVIDIA GeForce RTX 3090 and an Intel Core i7-14700K CPU, reaching convergence from scratch in about six hours.

TABLE I
Performance Comparison on a $10 \times 10 \times 10$ Bin Along With the Results of the Ablation Studies

Method	*Uti*	*Sta*	*Num*
Heuristic			
OnlineBPH [5]	51.6%	0.142	20.5
Best Fit [16]	57.9%	0.124	22.9
MACS [23]	50.6%	0.171	19.6
HM [21]	56.5%	0.105	22.1
DRL-based			
Zhao et al. [2]	70.9%	0.079	27.5
PCT [8]	72.7%	0.073	28.1
Xiong et al. [3]	<u>73.8%</u>	**0.068**	<u>28.3</u>
GOPT (ours)	**76.1%**	<u>0.070</u>	**29.6**
Ablation studies			
GOPT w/o PG	70.6%	0.086	27.5
GOPT w/o IR	73.2%	0.078	28.5
GOPT w/o PT	67.1%	0.085	26.2
GOPT-MLP	67.8%	0.079	26.4
GOPT-GRU	68.7%	0.082	26.9

Bold indicates the best and <u>underline</u> indicates the second best for that metric.

For experimental validation, we utilize the RS dataset [2] for training and evaluating our DRL agent. The bin dimensions $L \times W \times H$ are set to $10 \times 10 \times 10$, and the dimensions of items follow $\frac{\min(L,W,H)}{10} \leq l_t, w_t, h_t \leq \frac{\min(L,W,H)}{2}$. The dataset comprises 125 types of heterogeneous items, and sequences are dynamically generated by bootstrap sampling during training to reflect the variability in practical scenarios. An additional set of 1000 instances is generated for evaluation purposes, and the average performance is recorded.

B. Performance Evaluation

1) Baselines: To illustrate the superiority of our method, we select representative methods with publicly available implementations as baselines. We categorize these methods into two groups. The first group consists of four heuristic methods: OnlineBPH [5], Best Fit based on EP [16] that packs item in the lowest extreme point, MACS [23], and HM [21]. The second comprises three DRL-based methods: Zhao et al. [2], PCT [8], and Xiong et al. [3]. All methods are implemented and executed on the same desktop computer to ensure fair and rigorous comparisons. Furthermore, the DRL-based methods are trained from scratch with an equivalent number of steps, specifically 40 million, to eliminate training disparity bias.

2) Results: We evaluate the packing performance of these methods using three metrics: average space utilization of the bin (*Uti*), average number of packed items (*Num*), and standard deviation of space utilization (*Sta*), the latter of which assesses the stability of the methods across all instances. Quantitative comparisons, presented in Table I, demonstrate that our method outperforms all baselines in terms of *Uti* and *Num*. The findings indicate that our method achieves superior item packing and more efficient utilization of bin space. It is noteworthy that our method achieves the second-highest performance in terms of *Sta*,

TABLE II
GENERALIZATION PERFORMANCE ON BINS OF DIFFERENT DIMENSIONS

Method	Bin-10		Bin-30		Bin-50		Bin-100	
	Uti	*Num*	*Uti*	*Num*	*Uti*	*Num*	*Uti*	*Num*
Zhao et al. [2]	70.9%	27.5	72.4%	27.9	51.7%	20.6	/	/
Zhao et al. [10][1]	70.1%	27.1	71.7%	27.7	72.6%	28.1	71.3%	27.6
PCT [8]	72.7%	28.1	73.1%	28.1	70.1%	27.2	72.7%	27.9
Xiong et al. [3]	73.8%	28.3	75.6%	28.9	75.3%	28.8	73.8%	28.2
GOPT	**76.1%**	**29.6**	**76.0%**	**29.5**	75.7%	**29.4**	75.7%	29.4
GOPT (Bin-10)[2]	**76.1%**	**29.6**	**76.0%**	29.2	**75.8%**	29.2	**76.3%**	**29.6**

[1]Results are copied directly from [10] since the code is not available.
[2]GOPT (Bin-10) refers to the GOPT policy trained in Bin-10, which we directly apply to four environments to obtain testing results. In contrast, the other four methods, along with GOPT, require separate training and testing in these environments.

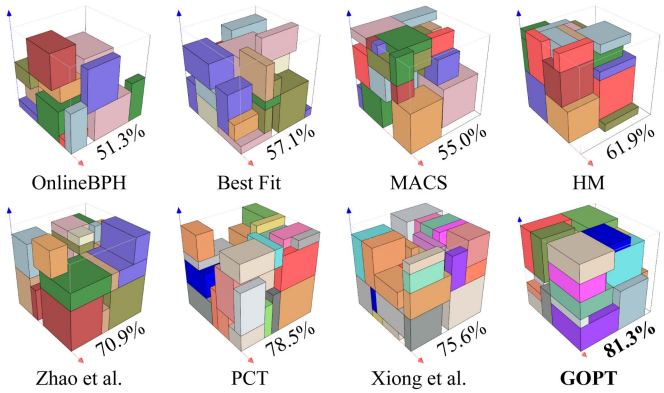


Fig. 4. Visualization results of different methods for an item sequence in a $10 \times 10 \times 10$ bin. The number beside each bin indicates the value of *Uti*.

with DRL-based methods showing comparable performance in this metric. Moreover, all DRL-based methods significantly outshine heuristic methods across all evaluation metrics. This advantage is attributed to the DRL-based method's ability to extract packing patterns and regularities from extensive training samples. In contrast, heuristic methods may struggle to generalize beyond their specific rules or strategies. The comparison with the baselines indicates our method's effectiveness. Furthermore, we depict the qualitative comparisons of visualized packing results from different methods in Fig. 4. It is observed that our results are visually superior to other competing methods.

C. Generalization

The capacity of learning-based methods to generalize across diverse datasets and unseen scenarios has consistently been a subject of scrutiny and interest. This section evaluates the generalization performance of our method across various bins of different dimensions and unseen items.

Generalization on different bins: In addition to the initial bin dimensions for the aforementioned training, we introduce three other environments where the bin dimensions are set to $30 \times 30 \times 30$, $50 \times 50 \times 50$, and $100 \times 100 \times 100$, respectively, and the item dimensions in the dataset are scaled up correspondingly. These environments are named Bin-10, Bin-30, Bin-50, and Bin-100. The search space for actions increases as the dimensions of bins grow, resulting in a higher complexity for finding a solution. To assess our method's generalization ability regarding the bin dimensions, we directly transfer our policy, trained solely in Bin-10, to the other three environments without fine-tuning. We additionally train and test our proposed GOPT, along with several DRL-based baseline methods [2], [3], [8], [10], separately in different environments for greater persuasiveness. The results in terms of *Uti* and *Num* are summarized in Table II. It is noted that Zhao et al.'s method [2] fails to converge in Bin-100. According to Table II, GOPT not only maintains consistent performance across different environments but also consistently outperforms other methods. Significantly, the policy GOPT (Bin-10) without retraining shows stable performance in environments divergent from the training one. Other DRL-based methods do not possess such ability as they need to be retrained when encountering varying bin dimensions. Intriguingly, some of them achieve relatively high performance in Bin-30. We surmise that this is due to a balance between the increased number of model parameters and the moderate problem complexity for this size, allowing for enhanced fitting capacity without the excessive difficulty observed at larger bins.

TABLE III
PERFORMANCE OF POLICIES TRAINED ON RS_{sub} WHEN EVALUATED ON RS_{sub} AND TWO DATASETS CONTAINING UNSEEN ITEMS

Method	RS_{sub}		RS		RS_{exc}	
	Uti	*Num*	*Uti*	*Num*	*Uti*	*Num*
PCT [8]	73.9%	28.0	73.7%	28.2	73.7%	29.3
Xiong et al. [3]	73.8%	27.9	73.0%	27.8	72.9%	29.0
GOPT	**75.5%**	**28.7**	**76.1%**	**29.5**	**75.7%**	**30.2**

Generalization on unseen items: Additionally, we conduct experiments to assess the generalization performance of our method using unseen items in Bin-10. This test is crucial and challenging as models often exhibit diminished performance when confronted with testing data that possess different characteristics. As previously mentioned, there are 125 distinct types of items in the RS dataset. We randomly exclude 25 types of items (RS_{exc}) from RS to train an agent with the sub-dataset RS_{sub} and test it with the complete RS and RS_{exc}. We select two baselines that performed well in previous experiments for comparison. As shown in Table III, our policy trained in the sub-dataset performs better than others when tested on both the full dataset RS and the dataset RS_{exc} consisting entirely of unseen items. This suggests

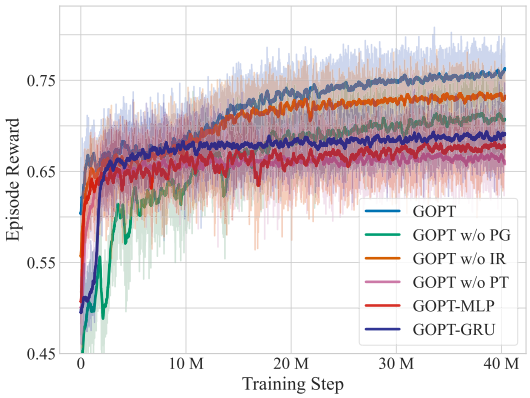


Fig. 5. Comparison of the training performance for the ablation studies. The results are obtained with 128 different random seeds.

the trained policy exhibits adequate generalization ability even on unseen items. We also observe an increase in *Num* across all methods on RS and RS_{exc}, likely due to these datasets having more small, easier-to-pack items.

D. Ablation Studies

Additional ablation studies are conducted to thoroughly analyze the impact of various components in our method. These components encompass the Placement Generator (PG), item representation (IR), and Packing Transformer (PT). We exclude PG and provide the neural network with all the placements and the corresponding masks to elucidate its effect. We also present results obtained without transforming the item representation from a three-dimensional vector to the proposed mode. Additionally, we conduct experiments by removing PT (GOPT w/o PT) and replacing PT with MLP (GOPT-MLP) and GRU (GOPT-GRU) to gain insights into its significance. The results are depicted in Table I. We also present reward curves versus training steps in Fig. 5.

As shown in Table I and Fig. 5, all three components introduced in this study exhibit favorable outcomes in line with our expectations. The comparative analyses indicate the performance of GOPT w/o PT, GOPT-MLP, and GOPT-GRU is significantly degraded compared to GOPT. It highlights the advantageous role of identifying spatial relations through the proposed PT in enhancing performance. This capability can be attributed to the superior efficacy of the attention mechanism in handling intricate sequential data and in learning long-range dependencies compared to other networks. Additionally, from Fig. 5, the models incorporating PT (GOPT, GOPT w/o PG, GOPT w/o IR) require more training data to achieve convergence than the models without PT (GOPT w/o PT, GOPT-MLP, GOPT-GRU), approximately 30 million versus 10 million. Besides, GOPT achieves greater space utilization and packs more items than GOPT w/o IR, indicating that the proposed item representation facilitates the DRL agent's learning and final performance. From Fig. 5, we note that GOPT w/o PG attains the least reward during the initial stages of training. This suggests that the PG module informed by human experience can contribute to improving sampling efficiency when the DRL agent has yet to accumulate substantial packing knowledge.

We also investigate the impact of reward design for the problem, encompassing the step-wise reward employed in this work, the terminal reward [31] defined as the final space utilization in an episode, and the heuristic reward [9] which adds a penalty term to avoid wasted space due to unreasonable actions. According to Table IV, the agent trained with the terminal reward shows the poorest performance, while the step-wise reward is more efficient despite its simpler and more intuitive nature than the heuristic reward.

TABLE IV
COMPARISON OF DIFFERENT REWARD FUNCTIONS

Reward designs	*Uti*	*Num*
Step-wise	**76.1%**	**29.6**
Terminal [31]	70.9%	27.6
Heuristic [9]	72.4%	28.0

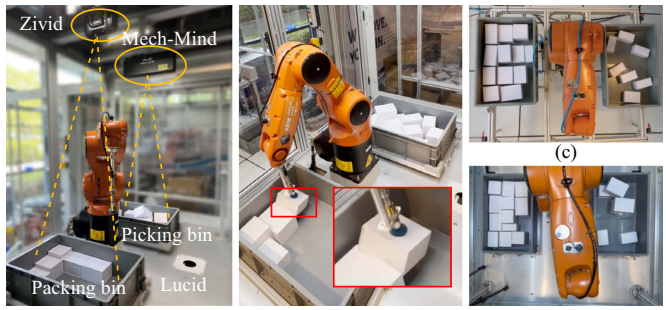


Fig. 6. The real-world experiments. (a) Our robot packing setup: A KUKA robot is equipped with a suction cup and three 3D cameras; (b) Failure case: The primary sources of failure in our experiments are measurement errors; (c) and (d) are snapshots of safe packing and tight packing.

E. Real World Experiment

We establish a physical robot packing testbed to verify the applicability of our method in the real world, as depicted in Fig. 6(a). The dimensions of the bin for packing items are 56 cm $\times$ 36.5 cm $\times$ 21 cm, which is discretized into a bin of $80 \times 52 \times 30$, with each cell measuring 0.7 cm in length. In this task, a robot selects a box from a bin, moves it within the Lucid camera's field of view to assess the box's dimensions and in-hand pose, and subsequently places it into another bin according to GOPT trained in the simulation. Meanwhile, two cameras are mounted to monitor these bins separately. The heightmap of the packing bin is generated through the segmentation and projection of the point cloud and the detection of rectangles. The pick-and-pack process proceeds until no boxes remain for picking or there is not enough space for packing the next box. Experiments show that a robot can utilize our method to complete the packing task in a real-world scenario. The demonstration video is provided in our supplementary materials.

From experiments, we observe that camera-induced measurement errors have the potential to cause collisions between boxes

during placement (see Fig. 6(b)). To prevent this, an additional 0.7 cm buffer space is allocated around each placed box, as shown in Fig. 6(c), resulting in an average space utilization of 67.5% across 20 tests. Reducing the buffer to zero increases the risk of errors and leads to 2 out of 20 tests failing, but achieves higher utilization (73.3% across 18 successful tests), as shown in Fig. 6(d). These findings provide an impetus for future research aimed at enhancing both system reliability in real-world robotic packing scenarios and the compactness of the packing outcomes.

V. Conclusion

We contribute a novel framework called GOPT for online 3D bin packing. GOPT embraces the Placement Generator module to generate placement candidates and represent the state of a bin with these candidates. Meanwhile, the Packing Transformer identifies spatial correlations for packing, which employs a cross-attention mechanism to fuse information from items and the bin effectively. Extensive experiments prove GOPT's superiority over existing methods, demonstrating notable enhancements not only in packing performance but also in generalization capabilities. Specifically, trained GOPT policy can generalize both across varying bins and unseen items. Finally, we successfully apply the trained packing policy in a robotic system, demonstrating its practical applicability. In the future, we plan to extend our method's application to include packing objects with irregular shapes, a common challenge in robotic pick-and-place tasks. We also plan to explore how to improve the reliability of the physical robot packing system.

References

[1] F. Wang and K. Hauser, "Dense robotic packing of irregular and novel 3D objects," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 1160–1173, Apr. 2022.

[2] H. Zhao, Q. She, C. Zhu, Y. Yang, and K. Xu, "Online 3D bin packing with constrained deep reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 1, pp. 741–749.

[3] H. Xiong, K. Ding, W. Ding, J. Peng, and J. Xu, "Towards reliable robot packing system based on deep reinforcement learning," *Adv. Eng. Inform.*, vol. 57, 2023, Art. no. 102028.

[4] O. X. d. Nascimento, T. A. d. Queiroz, and L. Junqueira, "Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm," *Comput. Operations Res.*, vol. 128, 2021, Art. no. 105186.

[5] 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 *Proc. Eur. Conf. Appl. Evol. Computation*, 2017, pp. 140–155.

[6] R. Verma et al., "A generalized reinforcement learning algorithm for online 3D bin-packing," 2020, *arXiv:2007.00463*.

[7] Z. Yang et al., "PackerBot: Variable-sized product packing with heuristic deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 5002–5008.

[8] H. Zhao, Y. Yu, and K. Xu, "Learning efficient online 3D bin packing on packing configuration trees," in *Proc. Int. Conf. Learn. Representations*, 2022.

[9] S. Yang et al., "Heuristics integrated deep reinforcement learning for online 3D bin packing," *IEEE Trans. Automat. Sci. Eng.*, vol. 21, no. 1, pp. 939–950, Jan. 2024.

[10] H. Zhao, C. Zhu, X. Xu, H. Huang, and K. Xu, "Learning practically feasible policies for online 3D bin packing," *Sci. China Inf. Sci.*, vol. 65, no. 1, pp. 1–17, 2022.

[11] S. Ali, A. G. Ramos, M. A. Carravilla, and J. F. Oliveira, "On-line three-dimensional packing problems: A review of off-line and on-line solution approaches," *Comput. Ind. Eng.*, vol. 168, 2022, Art. no. 108122.

[12] G. Dósa and J. Sgall, "First fit bin packing: A tight analysis," in *Proc. 30th Int. Symp. Theor. Aspects Comput. Sci. (2013). Schloss Dagstuhl-Leibniz-Zentrum Fuer Informatik*, 2013.

[13] G. Dósa and J. Sgall, "Optimal analysis of best fit bin packing," in *Proc. Int. Colloq. Automata, Languages, Program.*, 2014, pp. 429–441.

[14] L. Wang, S. Guo, S. Chen, W. Zhu, and A. Lim, "Two natural heuristics for 3D packing with practical loading constraints," in *Proc. Pacific Rim Int. Conf. Artif. Intell.*, 2010, pp. 256–267.

[15] S. Martello, D. Pisinger, and D. Vigo, "The three-dimensional bin packing problem," *Operations Res.*, vol. 48, no. 2, pp. 256–267, 2000.

[16] T. G. Crainic, G. Perboli, and R. Tadei, "Extreme point-based heuristics for three-dimensional bin packing," *Informs J. Comput.*, vol. 20, no. 3, pp. 368–384, 2008.

[17] F. Parreño, R. Alvarez-Valdés, J. M. Tamarit, and J. F. Oliveira, "A maximal-space algorithm for the container loading problem," *INFORMS J. Comput.*, vol. 20, no. 3, pp. 412–422, 2008.

[18] M. Agarwal, S. Biswas, C. Sarkar, S. Paul, and H. S. Paul, "Jampacker: An efficient and reliable robotic bin packing system for cuboid objects," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 319–326, Apr. 2021.

[19] A. Yarimcam, S. Asta, E. Özcan, and A. J. Parkes, "Heuristic generation via parameter tuning for online bin packing," in *Proc. IEEE Symp. evolving Auton. Learn. Syst.*, 2014, pp. 102–108.

[20] M. López-Ibáñez, J. Dubois-Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle, "The irace package: Iterated racing for automatic algorithm configuration," *Operations Res. Perspectives*, vol. 3, pp. 43–58, 2016.

[21] F. Wang and K. Hauser, "Stable bin packing of non-convex 3D objects with a robot manipulator," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 8698–8704.

[22] W. Shuai et al., "Compliant-based robotic 3D bin packing with unavoidable uncertainties," *IET Control Theory Appl.*, vol. 17, pp. 2241–2258, 2023.

[23] R. Hu, J. Xu, B. Chen, M. Gong, H. Zhang, and H. Huang, "TAP-Net: Transport-and-pack using reinforcement learning," *ACM Trans. Graph.*, vol. 39, no. 6, pp. 1–15, 2020.

[24] W. Kool, H. v. Hoof, and M. Welling, "Attention, learn to solve routing problems!," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=ByxBFsRqYm

[25] M. Nazari, A. Oroojlooy, L. Snyder, and M. Takác, "Reinforcement learning for solving the vehicle routing problem," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2018,vol. 31.

[26] Q. Que, F. Yang, and D. Zhang, "Solving 3D packing problem using transformer network and reinforcement learning," *Expert Syst. Appl.*, vol. 214, 2023, Art. no. 119153.

[27] O. Kundu, S. Dutta, and S. Kumar, "Deep-pack: A vision-based 2D online bin packing algorithm with deep reinforcement learning," in *Proc. 28th IEEE Int. Conf. Robot Hum. Interactive Commun. (RO-MAN)*, 2019, pp. 1–7.

[28] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba, "Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2017,vol. 30.

[29] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[30] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[31] J. Xu, M. Gong, H. Zhang, H. Huang, and R. Hu, "Neural packing: From visual sensing to reinforcement learning," *ACM Trans. Graph.*, vol. 42, no. 6, pp. 1–11, 2023.

[32] P. Li et al., "SelfDoc: Self-supervised document representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 5648–5656.

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

[34] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2015, *arXiv:1506.02438*.

[35] J. Weng et al., "Tianshou: A highly modularized deep reinforcement learning library," *J. Mach. Learn. Res.*, vol. 23, no. 1, pp. 12275–12280, 2022.