Robotics and Computer–Integrated Manufacturing 94 (2025) 102941

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Full length article

3D Vision robot online packing platform for deep reinforcement learning ☆

Xingyu Mu [a,b], Quanmin Kan [a,b], Yong Jiang [a,b], Chao Chang [a,b], Xincheng Tian [a,b,*], Lelai Zhou [a,b,*], Yongguo Zhao [c]

[a] Center for Robotics, School of Control Science and Engineering, Shandong University, Jinan 250061, China
[b] Engineering Research Center of Intelligent Unmanned System, Ministry of Education, Jinan 250061, China
[c] Institute of Automation, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250014, China

ARTICLE INFO

Keywords:
3D vision
Online hybrid palletizing
Deep reinforcement learning
Packing robot
Warehouse automation

ABSTRACT

In modern logistics and manufacturing, online mixed palletizing stands as one of the key automation technologies, facing challenges brought by the diversity of packages and real-time demand. However, traditional palletizing methods typically rely on preset rules, making them ill-suited to handle diverse bins in dynamic, real-time environments. This limitation becomes especially pronounced when dealing with complex palletizing tasks. To optimize the accuracy and operational efficiency of the palletizing process, this study, based on 3D vision technology and deep reinforcement learning techniques, designs a bin positioning algorithm utilizing projected bounding boxes. Additionally, spatial rotation position encoding is integrated into the decision-making process of the online palletizing network, designing a deep reinforcement learning algorithm for online mixed palletizing based on a masked attention mechanism. The paper also introduces a novel heuristic method—Boundary Point, which updates the palletizing state chain using key-point heuristics and "spatial" heuristics, and employs a pointer network for tail node selection. Experimental results demonstrate that the proposed method significantly improves average space utilization across the RS, CUT1, and CUT2 datasets. Finally, a 3D vision-based robotic online mixed palletizing experimental platform is designed and built, proving the effectiveness and application potential of the proposed algorithm.

1. Introduction

With the rapid development of the logistics industry and the rise of automated warehouses [1], there is an increasing demand for efficient and intelligent multi-specification item stacking and organizing solutions [2], in which automated palletizing technology is widely used. However, traditional palletizing algorithms usually rely on pre-set rules and fixed stacking patterns, making it difficult to cope with complex and changing production environments. For example: (1) Traditional robotic palletizing mainly targets single-specification boxes, resulting in poor real-time planning performance for multiple-specification boxes. (2) The position of box on the conveyor belt is unstable, and traditional algorithms depend on fixed pick-up positions, which can easily lead to pick-up failures. (3) The sequence and size of boxes arriving at the end of the conveyor belt are difficult to predict in advance, making pre-planning impossible.

In recent years, Online Mixed Palletizing has been gaining attention in academia and industry as a dynamic and flexible solution. Online Mixed Palletizing algorithms are designed to process bins of different sizes and shapes in real time, arranging them efficiently on pallets to maximize space utilization and stability. This process not only needs to take into account the physical properties of the delivery boxes themselves, but also must respond quickly to external changes, such as the dynamic adjustment of orders and the randomness of packages inflow. To effectively respond to real-time variations, 3D vision technology is applied for precise positioning and gripping of items within the palletizing system [3]. This technology facilitates accurate identification and extraction of essential information about the goods, serving as a foundational step to ensure the stability of the subsequent palletizing planning process. Since the palletizing problem is an NP-hard problem [4], it is difficult to find the optimal solution by an exact algorithm. Currently, scholars mainly deal with the online hybrid palletizing problem through some approximation algorithms, heuristic algorithms, and reinforcement learning algorithms. However, heuristic methods rely on manual packing experiences and lack flexibility in problem settings and the ability to learn. In contrast, deep reinforcement learning (DRL) has shown potential in solving combinatorial optimization problems [5]. Nevertheless, in the context of Online

☆ This work was supported by the Shandong Provincial Key Research and Development Program, China (2022CXGC020204, 2022CXGC010101), National Key R and D Program of China (No. 2022YFB4702500).

* Corresponding authors at: Center for Robotics, School of Control Science and Engineering, Shandong University, Jinan 250061, China.
E-mail addresses: txch@sdu.edu.cn (X. Tian), zhoulelai@sdu.edu.cn (L. Zhou).

https://doi.org/10.1016/j.rcim.2024.102941
Received 22 August 2024; Received in revised form 16 November 2024; Accepted 20 December 2024
Available online 15 January 2025
0736-5845/© 2025 Elsevier Ltd. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

3DBPP, the action space is extremely large and complex due to the various ways items can be placed in three-dimensional space. The DRL algorithm must find the optimal solution in this vast space, significantly increasing the computational complexity and convergence difficulty. Additionally, during the DRL training process, the reward signals may be very sparse. This means that the algorithm may need to perform many actions to receive meaningful feedback, resulting in a slower learning process and longer convergence times.

To address these challenges, this study investigates the online mixed palletizing of multi-specification boxes based on 3D vision and deep reinforcement learning. To accurately identify the position of the box and avoid the problem of deformed box damaging the palletized stacks during placement, this paper proposes a bin localization scheme. Building on the research by Zhao et al. [6], spatial positional encoding is introduced, and an online mixed palletizing method is implemented using a deep reinforcement learning approach based on masked attention mechanisms. Based on the concept of graph structures, we applied heuristic rules to systematically guide the generation of the palletizing state chain and introduced a new method for representing the placement points of bins during the palletizing process. Finally, a 3D vision-based online mixed palletizing experimental platform is constructed to verify the feasibility of the algorithm and analyze the results. The main contributions of this paper include:

1. This paper propose a point cloud feature-based algorithm for extracting bin information of the minimum enclosing box, which can effectively deal with the problem of dimensional error due to box extrusion and deformation, and reduce the risk of collision due to extrusion and deformation during the placement process.
2. Spatial rotational positional encoding is introduced into the palletizing state encoding network, and a masked attention network is proposed for encoding the palletizing state chain. By masking the attention weights between certain nodes, the network is ensured to focus more on the connections between important nodes, thereby improving spatial utilization and the number of packed items.
3. This study proposes a heuristic boundary point generation algorithm that combines the remaining space with key point positions. This approach ensures that during the generation of tail nodes, both the spatial layout of the box and the key point positions are comprehensively considered, leading to a more compact and efficient palletizing process.
4. For the online hybrid palletizing scenario, an experimental platform for online hybrid palletizing of robots based on 3D vision is designed to verify the practical application effect of the algorithm.

Through these studies, we hope to realize more efficient and intelligent solutions in online mixed palletizing tasks.

2. Related works

In 2D vision, image processing primarily relies on data analysis from single-plane images. In contrast, 3D vision generates high-precision three-dimensional information and models [7] through technologies such as depth sensors, stereo cameras, and LiDAR. As these technologies continue to mature, the application of 3D vision in industrial settings is expanding, especially in tasks like environmental perception for robotic systems, industrial measurement [8], object grasping [9], and welding [10]. It is becoming an important research focus in automation and robotics. Additionally, palletizing systems that integrate 3D vision demonstrate great potential in handling multi-specification items within dynamic environments [11], significantly enhancing operational flexibility and accuracy. Therefore, the application of 3D vision technology in extracting information from goods lays a solid foundation for further research on packing strategies.

The Bin packing problem has been a popular topic in combinatorial optimization. The most classic version, the one-dimensional bin packing problem (1D-BPP), aims to pack a set of items with varying scalar weights into multiple containers while minimizing the number of containers used [12]. Additionally, there are higher-dimensional variants, such as the 2D-BPP and 3D-BPP. The online 3D-BPP is derived from the offline 3D-BPP, and the insights gained from the offline studies help researchers to identify the relevant methods applicable to the online variant [13]. Earlier studies mainly treated 3D-BPP as an optimization problem to be solved [14,15], with a focus predominantly on its offline variant rather than the online one [16]. Many studies adopt approximation algorithms as a practical alternative. For example, Hifi et al. [17] proposed a mixed linear programming algorithm for 3D-BPP by relaxing the integer constraints in the problem. Furthermore, heuristic algorithms and reinforcement learning methods have gained wide attention in the application of 3DBPP.

2.1. Heuristic algorithms

Heuristic algorithms are problem-solving methods based on experience and intuition. While they do not guarantee finding an optimal solution, they often provide near-optimal solutions within a reasonable amount of time [18]. In the 3DBPP, heuristic algorithms employ a set of rules and strategies, based on experience, to search for effective packing schemes. These schemes aim to minimize gaps between items, maximize container space utilization, and reduce packing costs. Some heuristic methods attempt to pack boxes by defining specific rules. For instance, Martello et al. [19] proposed the use of corner points to determine the placement of items and defined an exact branch-and-bound algorithm for the 3D crating problem; Crainic et al. [20] introduced the concept of extreme points, placing items at these points to improve space utilization. Moreover, the extreme point rule is not dependent on a specific packing problem and can handle additional constraints, such as specifying the exact placement of items. Karabulut et al. proposed the Least Surface Area Heuristic algorithm [21], which first selects the deepest and most bottom-left vacant space as the location for the current item. Then, it fills this space as fully as possible to maximize the utilization of the container's internal space. Ha et al. were inspired by two basic heuristic algorithms and proposed the OnlineBPH algorithm to minimize the total cost of all used containers, and used the concept of EMS [22] to represent the free space in the container. Based on a special packing approach using packing indices, Wu et al. [23] introduced a method that combines genetic algorithms and the deepest bottom-left heuristic to solve the problem with variable container heights. Yarimcam et al. [24] proposed a generation of hyper-heuristics that differs from previous works using genetic programming or genetic algorithms to generate heuristics. They developed effective heuristic methods by adopting the Irace parameter tuning algorithm, providing online packing represented by a strategy matrix. Hu et al. [25] developed a Maximize-Accessible-Convex-Space strategy to optimize the available blank space to accommodate larger boxes that may arrive in the future. Wang et al. [26] proposed a method to minimize the height map, selecting the optimal placement location based on the heights of the already placed items to reduce the height differences in item stacking. By minimizing stacking height, they achieved a more compact and stable packing solution. Nguyen et al. [27] introduced an online bin-packing heuristic algorithm that leverages space partitioning and merging techniques. Chen et al. [28] introduced a new three-dimensional box size design and packing problem along with a hybrid optimization algorithm based on biogeography, aiming to maximize box volume utilization through box size optimization. Heuristic algorithms play an important role in solving 3DBPP, However, their reliance on empirically designed rules introduces certain limitations. Firstly, these rules are based on assumptions specific to particular scenarios, making it difficult for them to adapt effectively to changing real-world environments and requirements. Additionally, the rule design

process is often time-consuming and subjective, leaving it prone to biases and knowledge limitations of the designer. These limitations not only lead to instability in algorithm performance but also restrict their applicability.

2.2. Deep reinforcement learning methods

With the development of deep reinforcement learning technology [29], deep and reinforcement learning algorithms allow robots to autonomously learn and improve their creating strategies to better adapt to different crating scenarios and constraints. This approach holds the potential to overcome the limitations of heuristic algorithms, achieving more efficient and intelligent three-dimensional packing solutions. Hu et al. [30] divided the problem into cargo sequencing and placement decisions; Zhang et al. [31] adopted reinforcement learning in the placement decision process and reduced the dimensionality of the action space. Hu et al. [32] proposed an efficient transfer and placement network, using a precedence graph to represent the constraints of items originally stacked together. Yang et al. [33] introduced a method that uses heuristics to guide reinforcement learning, generating masks for priority placement locations based on heuristics. Que et al. [34] proposed a method to represent the states of containers and boxes separately, introducing planar features to capture the length and width information of containers.

Many methods require determining the order of items. However, in online 3DBPP (refer to Fig. 1), only one item's information can be observed at a time, and the sequence of items cannot be freely determined, rendering many offline packing methods inapplicable. Yang et al. [35] developed three heuristic algorithms and applied them to a DRL architecture, allowing agents to learn effective packing strategies in simulations. Murdivien et al. [36] used a game engine within a deep reinforcement learning framework to achieve a clear and intuitive visualization of the packing process and results, addressing the real-time sequential 3D container packing problem. Xiong et al. [3] proposed a candidate map mechanism to narrow down potential placement locations, alleviating the issues brought about by a large action space. Verma et al. [37] suggested a two-stage reinforcement learning-based approach for online scenarios, which was later refined by Zhao et al. [38] by decoupling the action space across different coordinate system directions. Due to the drawbacks of using height maps to represent container space, Zhao et al. [6] further introduced a new method for observing container space—the Packing Configuration Tree (PCT). Due to the complexity of the online packing problem, it is very laborious to construct a heuristic algorithm, and the method based on reinforcement learning has some advantages for the combinatorial optimization problem, which can adjust the network according to the reward feedback at that time, so as to be able to choose a more conducive way for the subsequent operation according to the different states, and thus to achieve a better coding effect.

3. Research on box positioning algorithm

In this section, we design a 3D vision-based method for cargo positioning. In Section 3.1, we first perform background removal using point cloud consistency, improved voxel downsampling, and statistical outlier removal as preprocessing steps. Then, in Section 3.2, the improved RANSAC algorithm is used to fit the plane, and in Section 3.3, the point cloud is projected onto the plane, incorporating the curved regions of the box edges into the box size. Finally, the boundary is extracted based on the point cloud density, the box coordinate system is established, and the minimum bounding box of the point cloud is obtained. The specific processing steps are shown in Fig. 5.

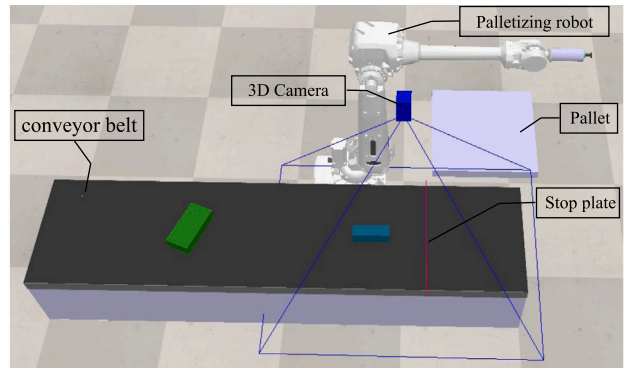


Fig. 1. Online 3D-BPP application scenarios in logistics, manufacturing, and warehousing.

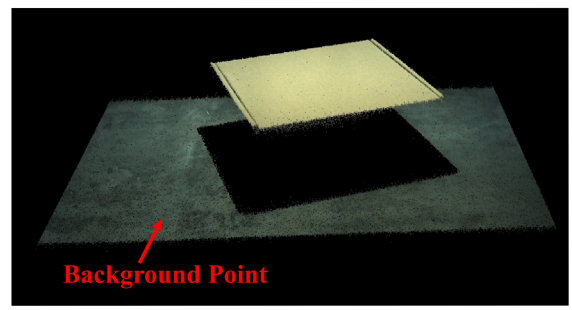


Fig. 2. Original package point cloud diagram.

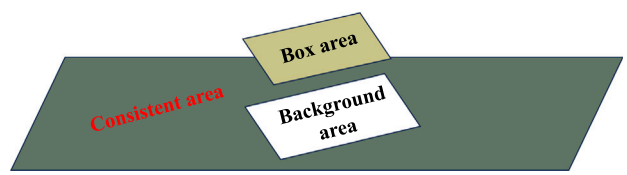


Fig. 3. "Background cancellation" diagram.

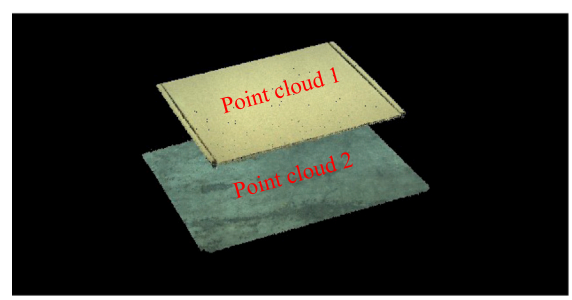
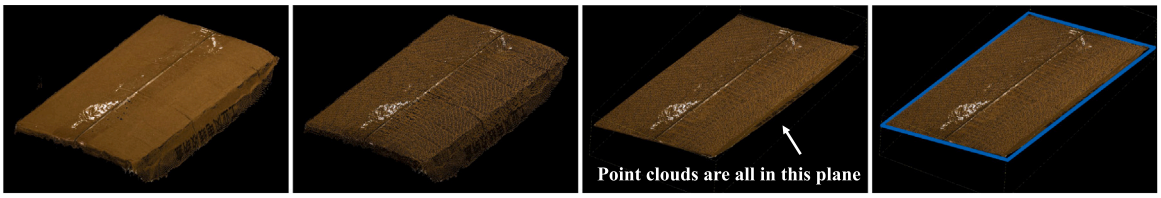


Fig. 4. Point cloud after removing the background.

3.1. Point cloud preprocessing

Due to the camera field of view and interference factors, the raw point cloud data often contain a lot of redundant background information, such as the point cloud of the external environment of the box (in Fig. 2). In order to effectively remove the environmental background interference, as shown in Fig. 3, we propose a preprocessing strategy called "Background cancellation", which eliminates the redundant environmental background information through consistent background removal from the point cloud. Since the camera position remains fixed, the environmental point cloud remains consistent across each capture.



(a) Origin point cloud (b) Downsample and filter (c) Fit the plane and project it (d) Get the bounding box

Fig. 5. The process of box localization algorithm. (a)~(d): The process of obtaining the minimum bounding box of a box deformed under compression in the box coordinate system.

Therefore, we first use the environmental point cloud without box as a reference base. Next, the point cloud containing the object is compared point-by-point with the reference baseline point cloud to identify and eliminate matching background points. After background removal, as shown in Fig. 4, the point cloud data containing only box information is obtained. Point Cloud 1 represents the surface points of the object, while Point Cloud 2 represents the background points obstructed by the object. By calculating the distance between these two planes, we can obtain the object's height information. Following this, we apply voxel downsampling and outlier statistical removal techniques. During the process of sampling under voxelization, we utilize an improved version of voxel downsampling, where the original point closest to the new point is selected to replace all points within the voxel, thus reducing data volume while preserving the original shape characteristics of the point cloud.

Subsequently, we apply Gaussian statistical filtering [39] to the point cloud data, a method that utilizes the statistical properties of the Gaussian distribution for effective filtering. This approach involves analyzing the statistical features within the neighborhood of each point and employing filtering operations based on these features to remove outliers and noise. First, each point p_i in the point cloud is traversed to calculate the mean μ_i and standard deviation σ_i of all points within the neighborhood N_i according to Eq. (1) and Eq. (2). Then, a threshold $(\mu - k\sigma, \mu + k\sigma)$ is set based on the mean and standard deviation, where k is a constant. If the value of p_i is outside the predetermined threshold range, it is considered an outlier and deleted. These steps are repeated for the point cloud until all points have been processed.

$$u_i = \frac{1}{|N_i|} \sum_{p_j \in N_i} p_j \tag{1}$$

$$\sigma_i = \sqrt{\frac{1}{|N_i|} \sum_{p_j \in N_i} (p_j - u_i)^2} \tag{2}$$

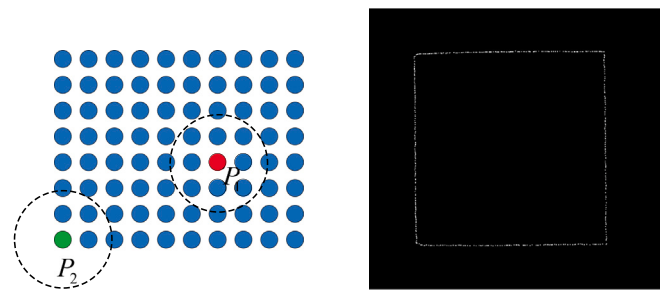
The preprocessed point cloud is shown in Fig. 5(b). It can be seen that the amount of point cloud data has been greatly reduced, while the main features of the upper surface of the box have been retained.

3.2. Plane extraction algorithm of the box

In order to get the size and shape information, we fit the plane of the box surface to get the size and shape information of the box. In this paper, We improved the traditional RANSAC algorithm [40] by performing a secondary fitting based on the initial plane fitting with RANSAC. That is, when the number of interior points of the model reaches a preset threshold, a planar model is re-fitted using these interior points.

3.3. Deformed box extraction method based on projected bounding box

In practice, as in Fig. 5(a), due to extrusion deformation and manufacturing, etc., the edges of the carton have certain curvatures, resulting in the length and width of the carton not coinciding with the length of the sides of the upper plane. These edge arcs can usually be captured by the camera as well. Accurately obtaining information about the size and position of the bins cannot be judged solely by the extracted planes. If the extruded deformation of the box is ignored, during the actual placing of the box, the depressed or protruding part of the extruded deformation may risk touching off or even damaging the boxes that have already been palletized in its vicinity. Therefore, this paper proposes a cargo information extraction algorithm based on planar least-surrounded box to reduce the average error of the size and position of boxes of different specifications, and to shorten the time of extracting information. First, we project the point cloud onto the extracted fitted plane, as shown in Fig. 5(c), the curved region of the box edge is also included in the calculation of box size. We use the boundary extraction method of point cloud point density [41] to extract the boundary of the projected point cloud based on the point cloud density, and construct the box coordinate system. On this basis, the least-surrounded box of the point cloud is calculated and inverse affine transformed back to the original camera coordinate system to obtain the accurate size and position information of the bins.



(a) Boundary extraction method (b) Boundary extraction result

Fig. 6. Boundary extraction.

We evaluate whether each point is on the edge of the point cloud based on the uniformity of the point cloud. Specifically, we can count the number of points within a radius R around each point. Theoretically, the number of points within a circle of radius R on a plane at a working distance H_i should be given by Eq. (3).

$$N_i = \rho_i \times \pi \times R^2 \tag{3}$$

If a point is located in the non-edge area of the point cloud, the number of points within its radius will be close to the theoretical number of points in the circular area. As shown in Fig. 6(a), in the non-edge area, the points are uniformly distributed within the circular area, and the number of points N_i is approximately equal to the area of the circle multiplied by the theoretical density ρ_i of the point cloud, such as the red point P_1 in the figure. However, if a point is located at the edge of the point cloud, certain areas within its radius may have no points, resulting in a number of points lower than the product of the density and the area of the circle, as in the green point P_2 in the figure. Based on this principle, by analyzing the point density around each point, the edge of the point cloud can be effectively extracted, identifying the edge regions within the point cloud. Fig. 6(b) shows the point cloud edges extracted using this method.

After extracting the edges of the box's top surface point cloud, we perform a linear fit to obtain the direction vectors of the lines and

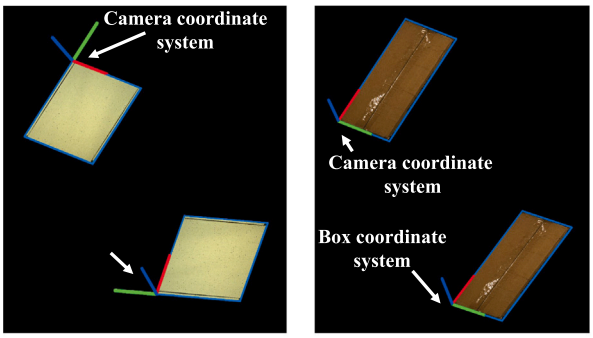


Fig. 7. Minimum bounding box transformation diagram. (a):Coordinate transformation of a regular and non-deformed box. (b):Coordinate transformation of a deformed box due to compression.

the coordinates of a point on each line. When the angle between the direction vectors of the two fitted straight lines is approximately $\pi/2$, the two straight lines are approximately perpendicular, i.e., they are the straight lines where the neighboring edges of the upper surface of the box are located. As shown in Fig. 7, a box coordinate system is established based on the direction vectors (x_1, y_1, z_1) and (x_2, y_2, z_2) of the two straight lines, and the normal vectors of the fitted planes (x_3, y_3, z_3) by using any point in the point cloud as the origin (o_x, o_y, o_z). According to the transformation matrix Eq. (4), the original point cloud is transformed into the box coordinate system, at which point the edges of the point cloud are parallel or perpendicular to the X-axis and Y-axis of the box coordinate system. Then, by calculating the minimum and maximum values of the transformed point cloud along the X-axis and Y-axis, the minimum bounding box is obtained (shown as the blue box in Fig. 5(d)). This minimum bounding box is transformed back to the camera coordinate system (shown as the blue box in Fig. 7), and the length, width, and pose of the box are determined based on the size and rotation direction of the bounding box. Note that the new coordinate system is an affine transformation, not a rigid transformation.

$$M = \begin{bmatrix} x_1 & x_2 & x_3 & o_x \\ y_1 & y_2 & y_3 & o_y \\ z_1 & z_2 & z_3 & o_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

4. Research on online mixed palletizing algorithm

For online mixed palletizing problems, researchers typically rely on constructive heuristics, which require specific rules for different scenarios and lack versatility. To address these limitations, in Section 4.1, this paper uses deep reinforcement learning to determine the code placement position, and formulates the online stacking problem as a Markov decision process. Specifically, we adopt the ACKTR (Actor–Critic using Kronecker-factored Trust Region) method proposed by Wu et al. in [42]. In Section 4.2, we propose a new heuristic algorithm to guide the generation of the tail node based on previous heuristic algorithms. In Section 4.3, we construct a reinforcement learning palletizing network using a masked attention network and embed spatial rotation position encoding into the network. In Section 4.4, a pointer network is employed to select appropriate tail nodes. The overall framework of the algorithm is shown in Fig. 8.

4.1. Markov decision process modeling for the palletizing problem

Reward Setting: Since the pallet size is fixed in this study, the larger the total volume of boxes on the pallet, the higher the space utilization rate of the pallet. Therefore, the reward in the reinforcement learning process is set to be related to the volume of each item placed. The reinforcement learning process aims to maximize the cumulative reward while also maximizing the space utilization of the pallet. Consequently, if an item is successfully placed at the front node, meaning the item is effectively stacked, the reward obtained is $r_t = c_r \times l_t \times w_t \times h_t$, where l_t, w_t, h_t represent the length, width, and height of the current box, respectively, and c_r is a constant serving as the scaling factor for the current item's volume. If the placement of an item fails, the reward returned is 0. By setting the reward to be positively correlated with the volume of the placed item, there is no need to discount future rewards in the reinforcement learning process, so the discount factor is set to $\gamma = 1$, treating future and immediate rewards as equally important.

Loss Function: The policy loss L_{policy} used in this study is shown in Eq. (5), and the value loss L_{value} is shown in Eq. (6). Here, r_t is the reward for the current decision, γ is the discount factor, $V(s_t + 1)$ is the discounted cumulative return, $V(s_t)$ is the value estimated by the critic network based on the current state chain encoding, $\pi\left(a_t \mid s_t\right)$ is the probability that the actor selects action a_t under state s_t. Both the policy loss and value function loss weights are set to 1.

$$L_{policy} = -\left(r_t + \gamma V\left(s_{t+1}\right) - V\left(s_t\right)\right) \log \pi\left(a_t \mid s_t\right) \tag{5}$$

$$L_{critic} = \left(r_t + \gamma V\left(s_{t+1}\right) - V\left(s_t\right)\right)^2 \tag{6}$$

4.2. Heuristic tail node generation method

The tail nodes represent the candidate locations where the current item can be placed. In order to obtain a quality palletizing strategy, the generation rules of the tail nodes are crucial. All possible placements are represented in the tail nodes, and heuristic algorithms are essentially summaries of human palletizing experience. Therefore, this paper adopts several popular heuristic algorithms for generating tail nodes and proposes a new heuristic algorithm, called the Boundary Point (BP), to guide the generation of tail nodes.

The EMS method only considers the four corner points of the maximum residual space and ignores other special locations, such as other special locations in the cargo upper plane. In this paper, BP is proposed, as shown in Fig. 9. When a bin is put into a container, a space is generated on each of its right sides, front sides, and upper sides. Since there is some overlap of the spaces, the figure is shown separately, where the brown boxes are the placed items, the blue parts are the EMS, the yellow points indicate the selected placement points in the EMS, and the red points are the additional boundary points, which are located at the corners of the boundary plane where the placed items overlap with the EMS space. Together, the yellow and red points form a set of boundary points.

The specific process of the BP algorithm is shown in *Algorithm* 1. The input is the incoming box $B_i(l, w, h)$, and the output is the current stacking position coordinates $BP(x, y, z)$. First, the remaining space is initialized, with the initial S_0 being the entire pallet space, and P being the set of BP, initially an empty set. Next, all corner points and special points at the bottom of the remaining space are collected. Then, the function $DRLOptimal(P)$ selects the optimal point from the collected point set through the tail node decision network as the placement point for the current box. The function $PlaceAction(P_i)$ executes the placement operation for the current item. Third, after the box is stacked and placed, the $Seg(S)$ function segments the remaining space based on the current maximum remaining space, and then the $Update(S)$ function updates the remaining space. Finally, any space that becomes a subset of another space is checked and merged, while also updating the special point set. The above steps are repeated each time a box is placed.

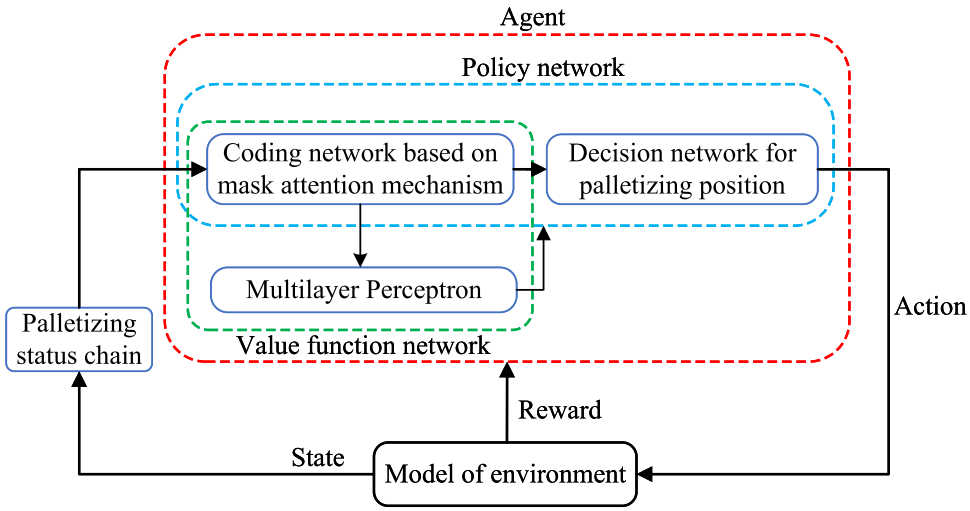


Fig. 8. Overall algorithm framework.

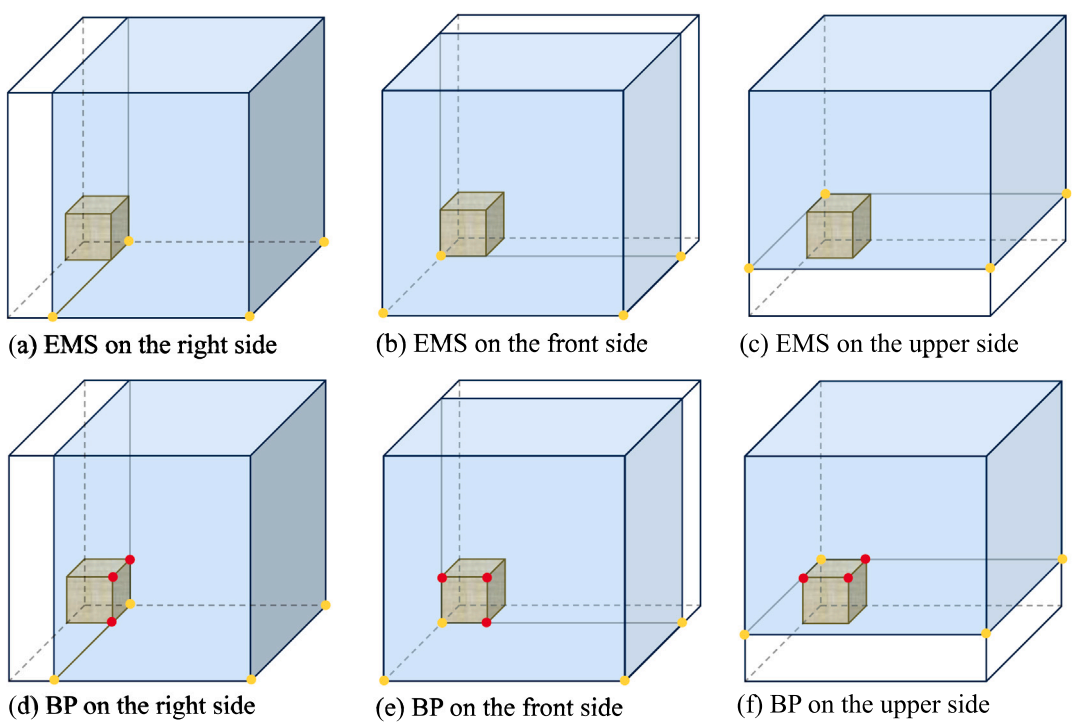


Fig. 9. BP generation diagram. (The brown box represents the placed package, with the blue space indicating Empty Maximal Space(EMS). (a)~(c): EMS selects placement points in the Empty Maximal Space on the right side, front side, and upper side. The method specifically selects the four corners of its bottom as placement points, represented by the yellow dots in the figure. (d)~(f): BP adds selectable placement points(red points) in the Empty Maximal Space on the right side, front side, and upper side. represented by the yellow and red dots in the figure.).

4.3. Reinforcement learning palletizing network

In order to fully utilize the information of the state chain, this paper adopts a graph neural network [43] to encode the state chain. Since the palletizing state chain changes gradually as the palletizing process proceeds, we employ the non-spectral graph attention network [44], which does not need to assume that the graph has a specific structure, to encode the palletizing state chain. When selecting tail nodes, we need to consider the influence of each tail node on the preceding nodes as well as the influence of each preceding node on the tail nodes. To address this issue, the paper employs a multi-head attention mechanism to evaluate the impact of each node on the others.

4.3.1. State chain encoding network design

As shown in Fig. 10, when a box arrives, potential placement positions are generated according to specific rules, referred to a “Nodes to be place” in the figure. One of these candidate positions is then selected for the placement of the current box. The placed boxes are termed “Placed nodes”. As the stacking process proceeds sequentially, the palletizing state chain is continuously generated.

In this paper, we design a coding network based on the mask attention mechanism for the palletized state chain as shown in Fig. 11, The front node F_t, the tail node T_t, and the current cargo node n_t of the state chain at time t are encoded by a multilayer perceptron (MLP) to convert the original position information into high-dimensional features. Each

Algorithm 1: BP Optimal algorithm

Input: Incoming Box $B_i(l, w, h)$
Output: Palletizing position $BP(x, y, z)$

Initialize the remaining space $S = S_0$;
$P \leftarrow \emptyset$;
for *Incoming box* $B_i(l, w, h)$ **do**
  foreach $S_i \in S$ **do**
    if S_i *can house the current container* **then**
      $P_c \leftarrow$ Add the Corner Point in S_i
      $P \leftarrow P + P_c$
    end
    foreach *Special point* $PS_i \in S_i$ **do**
      if PS_i *can house the current container* **then**
        $P \leftarrow P + PS_i$;
      end
    end
  end
  if $P \neq \emptyset$ **then**
    $BP(x, y, z) \leftarrow DRLOptimal(P)$;
    $PlaceAction(P_i)$;
  end
  $S_i \leftarrow Seg(S)$;
  Update(S);
end

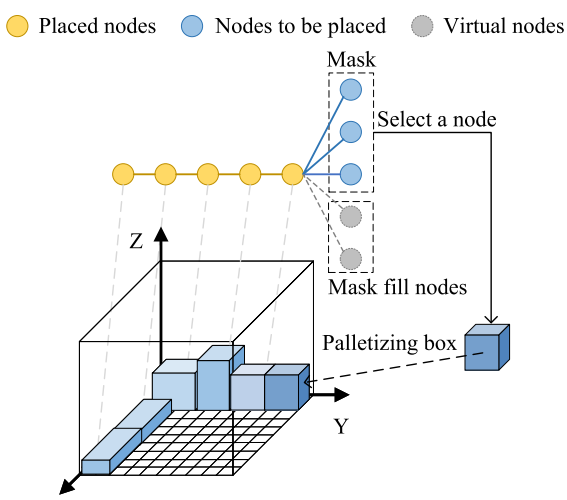


Fig. 10. Palletizing status chain.

MLP contains a hidden layer (32 neurons) and an output layer (126 neurons), utilizing the $LeakyReLU$ activation function to mitigate the vanishing gradient problem. Subsequently, the positional information is encoded using a spatial rotation positional encoding (3DPE) function. The encoded input is processed by a multi-head attention mechanism, in which the query Q, key K, and value V are calculated by the scaled dot product attention module [42] to obtain the attention output. The output from the multi-head attention mechanism is concatenated and then processed through a fully connected layer. Finally, the feedforward network ϕ_{FF} further extracts and maps features through two fully connected layers and $ReLU$ functions to complete the encoding network of the palletizing state chain as shown in Eq. (7):

$$h = \hat{h} + \text{Attention}(\hat{h}) + \phi_{FF}(\hat{h} + \text{ Attention }(\hat{h})) \tag{7}$$

In encoding the palletizing state chain, a masking operation is applied, as demonstrated by the Mask network layer in the top-left region of Fig. 11, where the masked attention mechanism focuses solely on the connections between the preceding node, the current object node, and the tail node, while ignoring secondary connections between tail nodes.

4.3.2. Spatial rotation position encoding

In the palletizing state chain of deep reinforcement learning, the formation of states and the modeling of the environment both rely on spatial position information, which impacts the quality of decision-making. There are challenges when processing spatial position information: the model may be insensitive to the position of objects in space, leading to a poor understanding of the environment's topology and spatial relationships, which reduces palletizing efficiency and accuracy.

To enhance the spatial position information in the palletizing state chain and improve the model's understanding of the environment, inspired by the explicit display of 3D relative distance information through the dot product of vectors in point cloud registration as in Lepard [45], we use the rotation position encoding proposed in [46] and extend it to 3D spatial positions, referred to as the 3D Position Encoder (3DPE).

Given a three-dimensional spatial position coordinate $P_i = (x, y, z) \in R^3$ and its corresponding feature $X_i^P = Embedding(l, w, h) \in R^d$, where X_i^P is the feature of the box (l, w, h) after $Input\ embedding$, we take d to be 126. The location encoding function $3DPE : R^3 \times R^d \rightarrow R^d$ is defined as in Eq. (8):

$$\mathcal{PE}\left(\mathbf{P}_i, \mathbf{x}_i^{\mathbf{P}}\right) = \Theta\left(\mathbf{P}_i\right)\mathbf{x}_i^{\mathbf{P}} = \begin{pmatrix} M_1 & & & \\ & M_2 & & \\ & & \ddots & \\ & & & M_{\frac{d}{6}} \end{pmatrix} \mathbf{x}_i^{\mathbf{P}} \tag{8}$$

where $\Theta\left(\mathbf{P}_i\right)$ is the block diagonal matrix. Each diagonal block of size 6×6 is defined by Eq. (9).

$$M_k = \begin{pmatrix} \cos x\theta_k & -\sin x\theta_k & 0 & 0 & 0 & 0 \\ \sin x\theta_k & \cos x\theta_k & 0 & 0 & 0 & 0 \\ 0 & 0 & \cos y\theta_k & -\sin y\theta_k & 0 & 0 \\ 0 & 0 & \sin y\theta_k & \cos y\theta_k & 0 & 0 \\ 0 & 0 & 0 & 0 & \cos z\theta_k & -\sin z\theta_k \\ 0 & 0 & 0 & 0 & \sin z\theta_k & \cos z\theta_k \end{pmatrix} \tag{9}$$

where $\theta_k = \frac{1}{10000^{6(K-1)/126}}, k \in [1, 2, \ldots, 126]$ encodes the index in the characteristic channel. Compared to the absence of positional encoding, its advantages are as follows:

1. By leveraging the orthogonal properties of $\Theta(P_i)$, spatial position encoding does not alter the length of the feature, only its direction. This helps stabilize the learning process of the palletizing state chain.
2. Relative 3D distance information can be explicitly represented through the dot product, that is expression $\left\langle \mathcal{PE}(\mathbf{P}_i, \mathbf{x}_i^{\mathbf{P}}), \mathcal{PE}(\mathbf{P}_j, \mathbf{x}_j^{\mathbf{P}}) \right\rangle$, which can be derived using Eq. (10):
$$\left[\Theta(P_i)x_i^P\right]^T \Theta(P_j)x_j^P = \left(x_i^P\right)^T \Theta(P_j - P_i)x_j^P \tag{10}$$
3. This enhances the spatial position information in the palletizing state chain, improving the model's understanding of the environment and the quality of decision-making.

4.4. Decision network for tail nodes

After the strategy network obtains the state code of the palletized stack in the tray, it needs to select a tail node as the location of the current box to be placed. As the palletizing process progresses, the palletizing state chain is continuously generated, the number of tail nodes also changes, resulting in an uncertain length of the input and

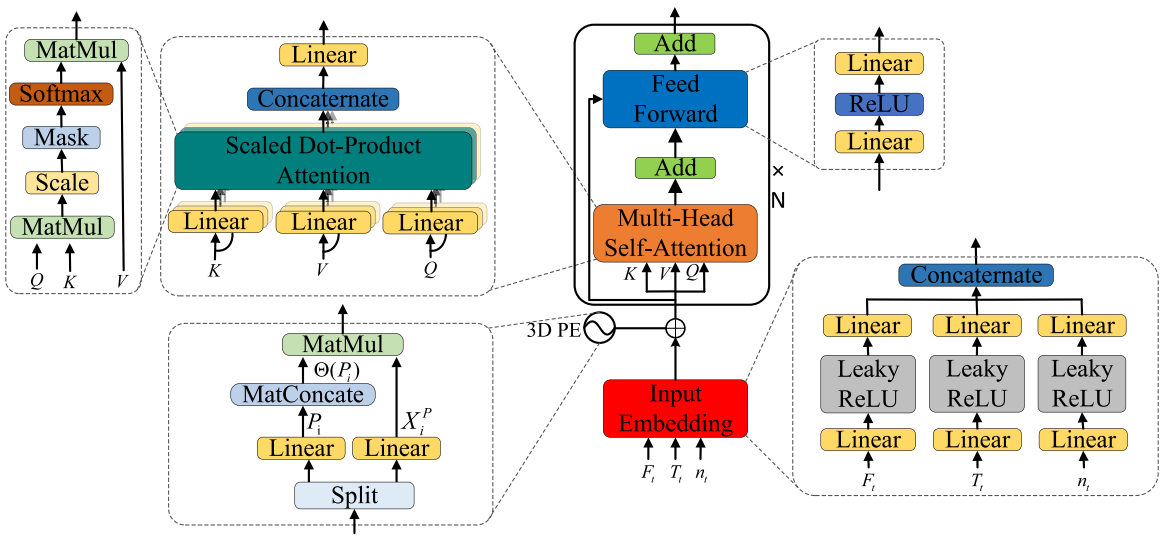


Fig. 11. Palletizing state chain encoding network. (The symbols are: Spatial rotation position encoding function (Section 4.3.2)). The preceding node of the state chain at time t is F_t, the trailing node is T_t and the current node is n_t(Section 4.3.1).

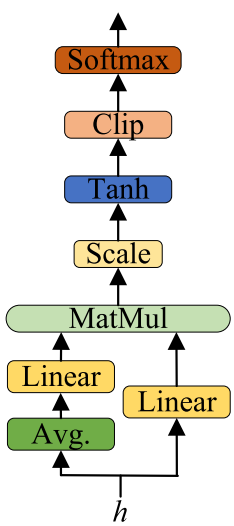


Fig. 12. Tail node selection network.

output. To solve this problem, this paper uses a pointer network [21] to select the appropriate tail node. The specific network structure is shown in Fig. 12. First, the average value of all encoded node features h is calculated to obtain the global feature. Then, a linear transformation is applied to the global feature to obtain the query, while the tail node features undergo a linear transformation to obtain the keys. Both linear layers have 64 neurons for input and output. Next, the association between each tail node and the query is computed by calculating the similarity between the query and keys through a dot product operation, with scaling applied to ensure numerical stability. The $Softmax$ function is then applied to convert the similarities into a probability distribution, yielding the probability of each tail node being selected. Based on Bello [47], the logical values of the nodes are clipped using an arctangent function, with the specific clipping range controlled by the hyperparameter C_{clip}. Ultimately, the entire tail node selection network can be represented by Eq. (11) and Eq. (12).

The value function network includes the encoding network of the code-palletizing state chain C_t and the value generation network, which maps the encoded state chain to a deterministic value. Specifically, the encoded nodes are averaged to obtain the global feature, which is then passed through a MLP to output the evaluation value of the entire state. Here, $W^q \in R^{126\times63}$, $W^k \in R^{126\times63}$ are projection matrices, d_k represents the dimension of the projected features.

$$\pi_\theta\left(T_t \mid C_t, n_t\right) = \mathrm{softmax}(D_t) \tag{11}$$

$$D_t = C_{\mathrm{clip}} \times \tanh\left(\frac{(W^q \mathrm{Avg}\,(h))^T\, W^k h}{\sqrt{d_k}}\right) \tag{12}$$

5. Experiment

5.1. Experiment description

This study employs simulation experiments and a specially designed robotic palletizing experimental platform to validate the effectiveness of the proposed algorithm. The robot online mixed palletizing system designed and constructed in this paper primarily includes a 3D vision system and a palletizing execution system, as shown in Fig. 13. The 3D vision system uses the PPN2000 3D camera from V-Sensor, with parameters listed in Table 1, to primarily collect point cloud data of the box and process it for size and pose details. The palletizing execution system uses an ABB IRB2600-20/1.65 type C robot, equipped with a pneumatic sponge suction cup with a diameter of 80 mm as the gripping actuator. The primary function of the palletizing execution system is to perform the picking and stacking operations. Communication between the computer and the robot is achieved through Socket communication. The overall system framework is illustrated in Fig. 14. During the agent's training process, we used the discrete dataset proposed by Zhao et al. [48], where the pallet size S^d is set to 10, with $d \in \{x, y, z\}$. The box size $s^d \in \mathbb{Z}^+$ does not exceed $S^d/2$ to avoid overly simplified scenarios. In the simulation experiments, the pallet dimensions are set to $10 * 10 * 10$. In the physical experiments, the pallet dimensions are set to $600 * 600 * 400$ mm^3. In the physical experiments of this study, due to the absence of a conveyor belt in the testing area, the boxes were randomly placed by hand within the camera's field of view.

5.2. Calibration process

5.2.1. Hand-eye calibration process

In this paper, the camera is configured in an "eye-to-hand" setup, as illustrated in Fig. 15, with the camera mounted externally on the robot. The primary method of hand-eye calibration involves the robotic arm holding a calibration board at its end-effector and moving the

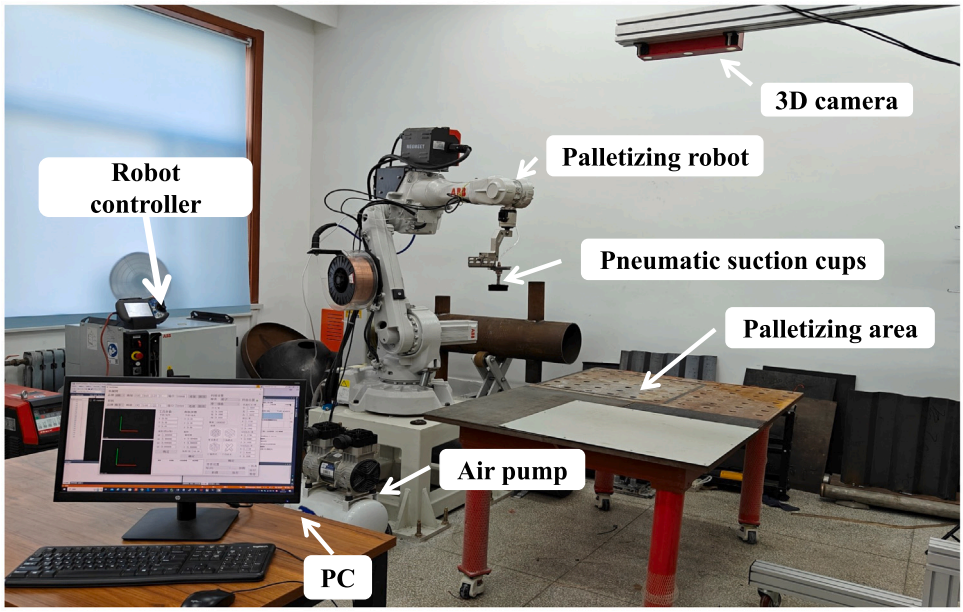


Fig. 13. Experimental platform. (The 3D vision system primarily captures images of the bins and processes the resulting point clouds, consisting mainly of a 3D camera and a PC. The palletizing execution system is mainly responsible for the grabbing and stacking operations of the bins, and consists primarily of a robotic system and a gripping actuator.).

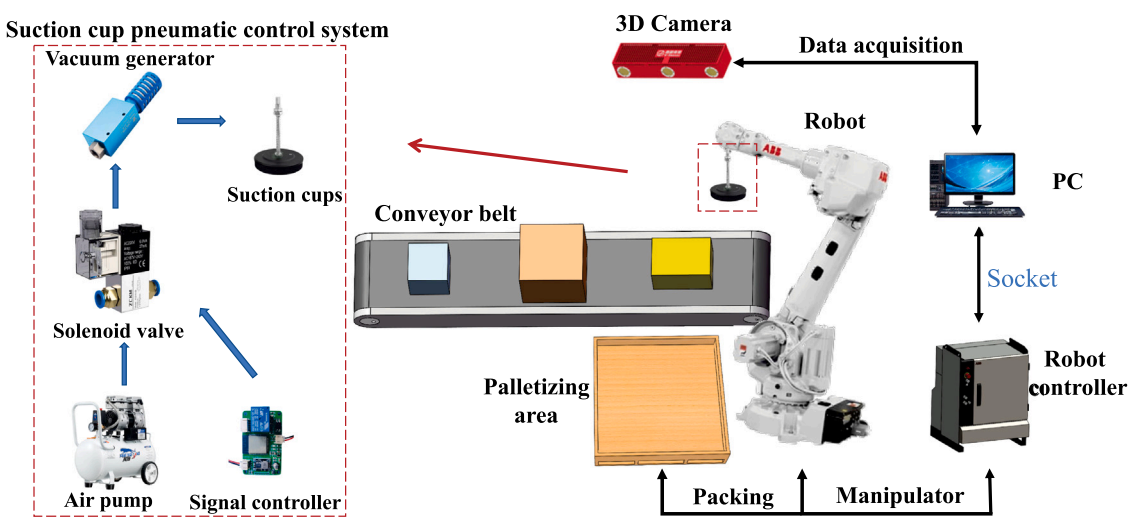


Fig. 14. Overall system platform.(The red box represents the control system of the pneumatic sucker).

Table 1
V-Sensor PPN2000 parameters.

Index	Parameter
Working principle	Binocular structured light
Working distance	1200–2000 mm
Near field of view	850×620 mm^2
Far field of view	1200×1050 mm^2
Resolution	1280×1024
Weight	1.4 kg
External dimensions	$326 \times 85 \times 57$ mm^3
Acquisition time	1.3–1.7 s

board within the camera's field of view to determine the transformation relationship between the camera coordinate system and the robot base coordinate system. Once the calibration board coordinate system is set, the camera captures the board, and based on the coordinates of points on the board, the transformation matrix from the calibration board coordinate system to the camera coordinate system ${}^{cam}_{cal}M$ is

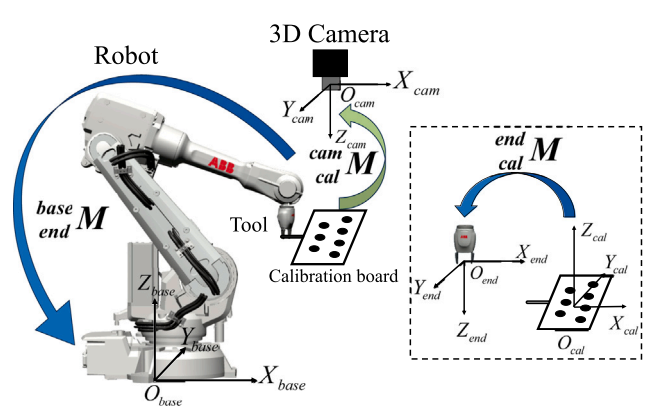


Fig. 15. Hand-eye calibration coordinate conversion diagram.

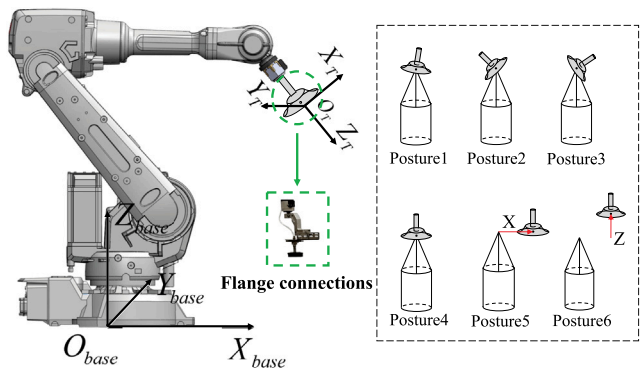


Fig. 16. Grasp actuator calibration diagram.

calculated. The calibration board is held at the end of the robotic arm by a tool gripper, maintaining a fixed relative position with the arm's end effector. This fixed transformation matrix is represented as ${}^{end}_{cal}M$. The transformation matrix from the arm's end effector to the robot base is labeled ${}^{base}_{end}M$, and the transformation matrix from the camera coordinate system to the robot base coordinate system is denoted as ${}^{base}_{cam}M$. During two consecutive movements of the robotic arm, ${}^{end}_{cal}M_1 = {}^{end}_{cal}M_2$ holds true. There is ${}^{end}_{cal}M = {}^{end}_{base}M \cdot {}^{base}_{cam}M \cdot {}^{cam}_{cal}M$. Setting $A = {}^{base}_{end}M_2 \cdot \left({}^{base}_{end}M_1\right)^{-1}$, $X = {}^{base}_{cam}M$, and $B = {}^{cam}_{cal}M_2 \cdot \left({}^{cam}_{cal}M_1\right)^{-1}$ allows us to apply the equation $AX = XB$. Solving [49] for X yields the transformation matrix ${}^{base}_{cam}M$, which represents the conversion from the camera coordinate system to the robot base coordinate system. Eventually we solve for

$$ {}^{base}_{cam}M = \begin{bmatrix} 0.9943 & -0.0046 & 0.1063 & 1020.93 \\ -0.002 & -0.9998 & -0.016 & -388.91 \\ 0.1064 & 0.01581 & -0.994 & 1589.06 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

5.2.2. TCP calibration process

The pneumatic suction cup serves as the gripper actuator in the robotic palletizing system and is mounted at the end of the robotic arm. To achieve accurate grasping, it is essential to determine the transformation relationship between the center of the suction cup (i.e., the end of the gripper actuator) and the robot flange coordinate system. Given that ABB robotic arms support tool coordinate calibration, Fig. 16 illustrates the calibration process. In this study, the four-point + XZ method is used to calibrate the gripper actuator's tool coordinate system, with Posture5 and Posture6 defining the X and Z axes of the gripper actuator coordinate system, respectively, and their cross product defining the Y axis. The final transformation parameters obtained between the gripper actuator tool coordinate system and the robotic arm flange coordinate system are: translation vector $t = [-122.1051\ \text{mm}, -1.493982\ \text{mm}, 441.3829\ \text{mm}]$ and *Quaternion* $q = [0.99473, -0.01383, -0.01480, -0.10041]$.

5.3. Feasibility of box positioning algorithm

The preprocessed point cloud undergoes further processing to extract the size and pose information of the boxes. Although accurately determining the pose of the items can be challenging, the method we use for obtaining the size of the items is closely related to their pose. When the size is accurately obtained, the pose is also relatively accurate. Therefore, it is essential to verify the accuracy of the adopted method for obtaining item information, with a particular focus on the size and position data.

The position information of the cargo is obtained by marking the center point of the upper surface of the box, and then manually picking up the marked point in the point cloud to obtain the actual position information of the bin. We have conducted experiments on a variety of cargo sizes, and the error maps of the box size and position are shown in Fig. 17. Positive values indicate that the calculated value is greater than the actual value, while negative values indicate that the calculated value is smaller than the actual value.

It can be seen from Fig. 17(a) that the average error of the box size acquisition scheme for different box sizes is less than 0.2 mm, and from Fig. 17(b) that the average error of the box position acquisition in this paper is less than 0.15 mm. The vertical axis in the figures represents the mean error for each type of box, and the error bars display the range of errors. The size error is within 0.25 mm, with the smallest error in height, while the position error is within 0.15 mm, with the smallest error in the Z direction. This indicates that the error threshold of the algorithm is sufficient to meet the palletizing requirements of industrial sites.

Overall, the positioning algorithm proposed in this paper is sufficient for the accuracy required for subsequent palletizing, with minimal error ranges in extracting box information, which indicates that the algorithm in this paper is relatively stable in processing the box point cloud. As shown in Fig. 17(c), the running time of the box information acquisition scheme is less than 0.5 s, which meets the real-time requirements of online palletizing.

Table 2
Comparison of box size error.

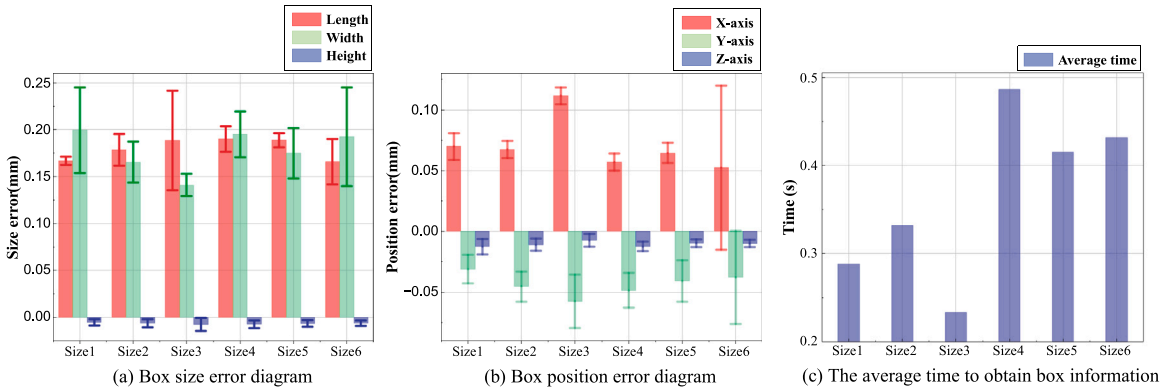
Methods	Box1 Error/mm			Box2 Error/mm		
	Length	Width	Height	Length	Width	Height
Bae et al.	≤ 2.0	≤ 8.0	≤ 1.0	/	/	/
Peng et al.	/	/	/	≤ 0.40	≤ **0.30**	≤ 0.40
Ours	≤ **0.25**	≤ **0.32**	≤ **0.22**	≤ **0.31**	≤ 0.33	≤ **0.16**

5.4. Comparison of recognition error thresholds

Most studies on palletizing primarily focus on optimizing box placement to improve space utilization and stacking efficiency, with less attention given to the measurement, positioning, and grasping of incoming boxes. Therefore, we selected some methods specifically designed for measuring box dimensions for comparison. For example, Bae et al. [50] used a 2D camera to capture product images and employed an image segmentation model to determine the pixel area of the product. A TOF sensor was then used to measure the height, and a height-based conversion equation was applied to convert pixel units to millimeters, calculating the horizontal and vertical lengths of the object. Peng et al. [51] measured dimension information using laser triangulation and deep learning, utilizing two laser box images from a camera and cross-line laser projector. Through laser triangulation, they calculated the precise geometric shape of the box from the three-dimensional coordinates of key points in the laser box images. In the experiment, Box1 has dimensions of 270 × 180 × 150 mm, and Box2 has dimensions of 550 × 480 × 380 mm. Table 2 shows our comparison results with these two methods. Although the algorithm by Peng et al. performs slightly better than ours in terms of error thresholds for width measurement, our method outperforms in other aspects. This result confirms that our proposed method for acquiring box information effectively meets the application requirements of industrial palletizing and provides a reliable pre-processing foundation for subsequent palletizing stages.

5.5. Comparison with other algorithms

To verify the effectiveness of the online stacking algorithm used in this paper, several sets of experiments were designed to verify it. The data used in the experiments were proposed by Zhao [48], with the datasets named RS, CUT-1 and CUT-2. Fig. 18 shows the Palletizing results of our method on the three datasets.



(a) Box size error diagram (b) Box position error diagram (c) The average time to obtain box information

Fig. 17. Experimental results of Different kinds of boxes information.

Table 3
Comparison of different tail generation rules.

Method	RS dataset		CUT-1 dataset		CUT-2 dataset	
	Space utilization (%)	Number (unit)	Space utilization (%)	Number (unit)	Space utilization (%)	Number (unit)
CP	68.511	26.276	70.911	17.872	70.534	18.422
EP	68.376	26.310	64.897	16.418	65.319	17.020
EMS	78.044	30.066	77.340	19.580	78.045	20.522
FC	74.519	28.868	70.818	17.996	71.742	18.932
BP	**78.871**	**30.416**	**77.748**	**19.660**	**78.264**	**20.558**

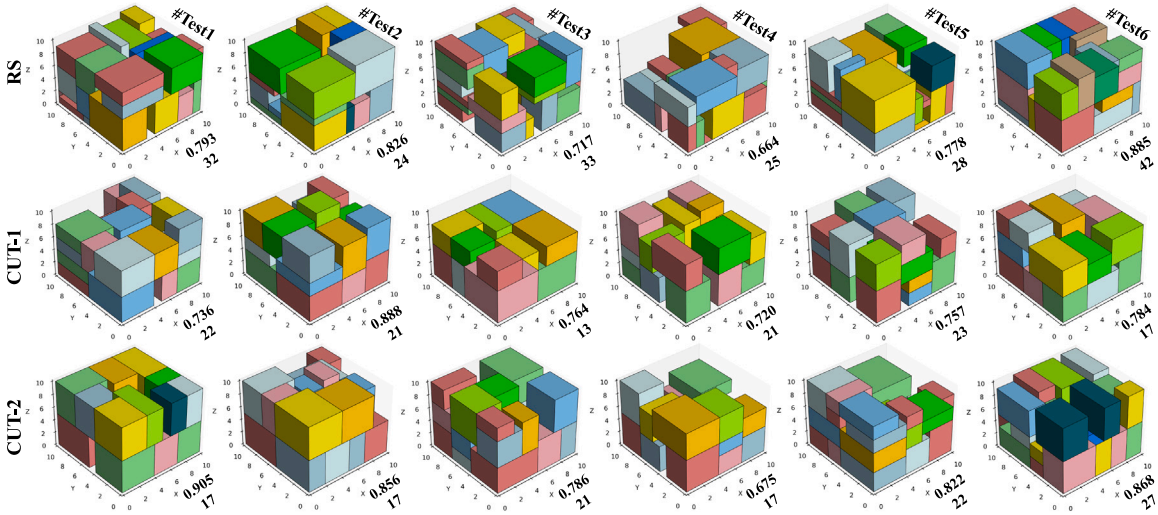


Fig. 18. Visualize packaging results using three different datasets (RS,CUT-1 and CUT-2). The values next to each subgraph are space utilization and number of palletizes.

5.5.1. Comparison of tail node generation rules

Different tail node generation rules create different palletizing state chains. We conducted comparative experiments on tail nodes generated by different heuristic rules, with the results presented in Table 3. The data indicate that compared to traditional heuristic rules such as CP, EP, and EMS, the BP method proposed in this paper significantly improves space utilization when generating tail nodes. This implies that the tail nodes generated using the BP method can more effectively fill the space, minimizing empty regions during the stacking process, thus achieving more compact and efficient palletizing. This improvement is attributed to BP's combination of remaining space with key points, considering both the spatial layout of the bin and the key point positions during the tail node generation process.

5.5.2. Comparison of online palletizing algorithms

In order to validate the performance of the deep reinforcement learning method used in this paper, we compared our algorithm with several other online palletizing algorithms. The comparative algorithms include the Least Surface Area Heuristic (LSAH) [30], Deepest Bottom Left with Fill (DBL) [21], Maximize Accessible Convex Space (MACS) [25], Online Bins Packing Heuristic (OnlineBPH) [22], Heightmap Minimization Heuristic (HM) [26], Boundary Rule (BR) [48], Random, Online 3D Bin Packing with Constrained Deep Reinforcement Learning (CDRL) [48], HIDRL [35] and the PCT method.

In the experiments, space utilization and the number of stacked items were used to evaluate the algorithms' performance. The experimental results represent the average values from 500 experiments, as detailed in Table 4. The experimental data indicates that heuristic algorithms tend to perform relatively poorly in bin packing problems. LSAH selects placement positions by minimizing surface area, but due to its lack of rotation optimization for items and the simplicity of its spatial utilization strategy, it performs only moderately in terms of space utilization and packed item count. DBL includes some optimization in

Table 4
Comparison of the effects of different algorithms in online palletizing scenarios.

Algorithm	RS dataset		CUT-1 dataset		CUT-2 dataset	
	Space utilization (%)	Number (unit)	Space utilization (%)	Number (unit)	Space utilization (%)	Number (unit)
LSAH	52.682	20.996	58.403	15.228	57.183	15.336
DBL	60.007	23.736	67.760	17.418	66.721	17.798
MACS	57.623	22.528	61.544	15.798	62.239	16.610
OnlineBPH	51.590	20.544	66.081	16.928	62.446	16.734
HM	56.575	22.414	62.567	16.016	63.703	16.966
BR	48.709	19.566	56.696	14.778	57.492	15.486
Random	37.134	15.162	40.397	10.848	41.722	11.492
CDRL	48.560	11.774	74.560	18.956	65.530	17.318
PCT	72.955	28.412	75.989	19.278	75.903	20.032
HIDRL	73.922	28.782	76.681	19.359	77.162	20.289
Ours	**78.871**	**30.416**	**77.748**	**19.660**	**78.264**	**20.558**

placement selection but focuses solely on deep filling in the bottom-left corner, which can lead to wasted space, particularly when item shapes and sizes vary. MACS emphasizes expanding accessible convex spaces and attempts to optimize space utilization through a convex polyhedral structure. However, this approach has significant limitations in handling complex spatial structures, leading to performance that is inferior to deep reinforcement learning methods. OnlineBPH represents remaining space through EMS and selects the placement with the smallest remaining space. Although this strategy can improve single-placement utilization, it is less effective for overall spatial usage. The HM algorithm optimizes space utilization by minimizing stack height differences; however, its ability to handle complex stacking structures is limited, making it difficult to achieve the effectiveness of deep reinforcement learning methods. BR aligns placements with current box boundaries, focusing on boundary alignment and stability, which suits simple packing tasks but performs poorly in complex 3D scenarios. Random places items by randomly selecting positions that meet placement conditions, resulting in poor performance, which indicates that simple randomization strategies struggle to utilize space effectively. Heuristic algorithms generally require specialized designs for different types of problems or conditions, which limits their versatility.

The methods based on deep reinforcement learning are more effective in addressing online palletizing challenges. Among the deep reinforcement learning algorithms, the space configuration-based PCT algorithm demonstrates superior palletizing performance compared to the discretized space-based CDRL method. This suggests that the approach of constructing remaining space is advantageous for deep reinforcement learning methods in online palletizing, leading to enhanced outcomes. Furthermore, the integrated heuristic HIDRL algorithm, which incorporates real-world physical principles like the laws of physics and human packing strategies into the DRL framework, also exhibits significant advantages in bin packing problems. The experimental results show that the space utilization and the number of pallets achieved by the method implemented in this study exceed those of other methods across three datasets. This demonstrates that the proposed method can more effectively enhance space utilization in online palletizing problems by combining deep reinforcement learning with space configuration strategies.

5.6. Ablation study

In order to evaluate the impact of each module on system performance, this study uses a deep reinforcement learning model without any modules as the baseline model (Baseline). By gradually adding the 3DPE, BP and Mask modules, the impact on space utilization and the number of stacked objects is observed. As shown in Fig. 19, We plotted the learning curves of spatial utilization in the ablation study. The steeper ascent of the curves suggests that adding modules accelerated the learning process, enabling the agent to learn box-packing strategies more efficiently.

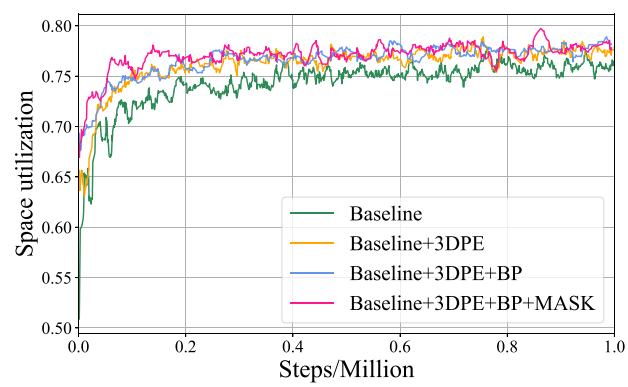


Fig. 19. Comparison of the training performance of the ablation study. The curves represent the learning progress of the Baseline and the repeated training processes with added modules.

The experiments were conducted on three datasets (RS, CUT-1, and CUT-2). As shown in Table 5, after adding the 3DPE module to the Baseline model, the average space utilization on the RS, CUT-1, and CUT-2 datasets increased by 2.305%, 0.564%, and 0.996%, respectively, and the average number of stacked items increased by 0.862, 0.194, and 0.246, which is due to the fact that the position coding module enhances the model's understanding of spatial relationships by introducing more positional features, thus increasing the space utilization and the number of palletizing boxes.

The system performance continues to improve with the further addition of the BP module, as the BP method combines remaining space with key points, providing more options for generating tail nodes. This optimization of boundary point position and quantity during tail node generation enhances packing efficiency and reduces space waste.

Finally, the addition of the Mask module, on top of the 3DPE and BP modules, resulted in the best system performance. Because the mask attention can better focus on the important connections between key nodes, which enables the algorithm to learn the interactions between the nodes more effectively, reduce unnecessary calculations, and improve the generalization ability and computational efficiency of the model.

Compared to the baseline model, the final model improved average space utilization by 3.149%, 1.204%, and 1.946%, respectively, and increased the average number of stacked items by 1.168, 0.32, and 0.454 on the three datasets.

In conclusion, by gradually introducing the 3DPE, BP algorithm, and Mask module, the system achieved significant improvements in space utilization and stacking quantity across all three datasets. This

Table 5
Ablation studies on three datasets.

Baseline	3DPE	BP	Mask	RS dataset		CUT-1 dataset		CUT-2 dataset	
				Space utilization (%)	Number (unit)	Space utilization (%)	Number (unit)	Space utilization (%)	Number (unit)
✓				75.722	29.248	76.544	19.340	76.318	20.104
✓	✓			78.027(↑2.305)	30.11(↑0.862)	77.108(↑0.564)	19.534(↑0.194)	77.314(↑0.996)	20.350(↑0.246)
✓	✓	✓		78.169(↑0.142)	30.170(↑0.060)	77.340(↑0.232)	19.616(↑0.082)	77.608(↑0.294)	20.430(↑0.080)
✓	✓	✓	✓	**78.871(↑0.702)**	**30.416(↑0.246)**	**77.748(↑0.408)**	**19.660(↑0.044)**	**78.264(↑0.656)**	**20.558(↑0.128)**

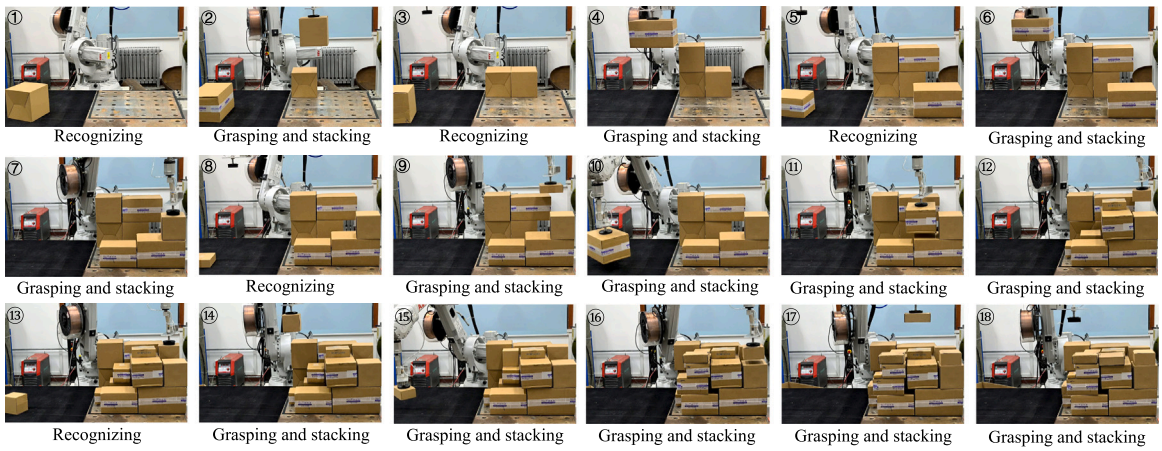


Fig. 20. Palletizing experiment process visualization. Recognize shows that the camera takes pictures of the box and obtains the size and position information of the box; Grab and Carry show the process of the robot moving the box over the tray and placing it.

Table 6
Dimensions of the experimental box.

Num	Size/mm	Num	Size/mm
#1	197*197*197	#9	160*235*120
#2	292*194*175	#10	155*155*53
#3	197*142*110	#11	147*110*85
#4	127*127*125	#12	208*208*55
#5	200*200*400	#13	107*107*65
#6	217*142*110	#14	94*132*85
#7	184*262*155	#15	66*66*23
#8	194*292*400	#16	55*105*23

demonstrates that these modules play an important role in improving the performance of the system, and the addition of each module has had a positive effect on the improvement of the system.

5.7. Comparison of practicality of packing robot

In practical palletizing scenarios, the 3D camera scans the bins and transmits the resulting point cloud to the computer. The computer processes the point cloud to obtain information on the size and pose of the bins, and plans the grabbing and stacking positions. It then transmits the relevant bins information to the palletizing execution system. As shown in Fig. 20, the entire process of recognizing, grasping, and then stacking the boxes is illustrated.

This paper aims to replace manual operations in online mixed palletizing scenarios with a robotic system. To this end, we designed comparative experiments involving operators with backgrounds in science or engineering, all holding a master's or doctoral degree. The experiment used 16 different types of boxes varying in quantity and size, randomly divided into 9 groups for testing. Table 6 provides an overview of the dimensions of each box type.

The results were compared based on average space utilization and the number of items. As shown in Table 7, the algorithm proposed in this paper outperformed human operations in certain cases, but the average space utilization was slightly lower than that of human operators. This may be because, during manual palletizing, items can be placed more tightly, whereas robotic arms need to leave gaps to avoid collisions. Overall, the robotic palletizing system demonstrated significant potential and advantages in practical applications.

5.8. Discussion of online palletizing methods

As shown in Table 8, there are differences among various methods in terms of dependency, generalizability, spatial utilization, rule design, and spatial location perception abilities.

Online heuristic methods typically rely on fixed rules and manual expertise, resulting in weaker generalizability. Their rule design is often optimized for specific datasets, performing poorly on other datasets and struggling to adapt to new application scenarios. Furthermore, these methods tend to perform poorly in online palletizing scenarios, often leading to significant space wastage.

In contrast, DRL-based methods and our proposed approach rely less on manual expertise and demonstrate strong generalizability across multiple datasets. Our method introduces rotational positional encoding, which greatly enhances spatial location perception abilities, effectively reducing space wastage and further improving spatial utilization. In terms of adaptability to dynamic environments, both DRL-based methods and our approach show higher flexibility, dynamically adjusting decisions based on environmental changes, whereas heuristic methods lack this adaptive capability.

6. Conclusion

This research is based on 3D vision technology and deep reinforcement learning technology to study the online mixed palletizing problem. A novel algorithm for bin information extraction based on point cloud features and the minimum bounding box is proposed. By considering the actual curvature and deformation of the box edges, the algorithm reduces collision risks due to deformation during placement and decreases the average error in extracting the size and position of box, thereby improving the overall efficiency and accuracy of the palletizing system.

Table 7
Comparison between oursystem and other palletizing methods.

Method	Metric	List1	List2	List3	List4	List5	List6	List7	List8	List9	Average
Human operation	Utilization (%)	**67.016**	66.021	69.821	72.928	77.564	**79.895**	77.032	65.216	**80.293**	72.865
	Number (unit)	**40**	25	24	31	24	**32**	25	21	**26**	27.556
Our palletizing system	Utilization (%)	65.248	**67.538**	67.056	**74.445**	**78.135**	74.092	71.155	**66.101**	60.662	69.381
	Number (unit)	38	**27**	20	**33**	**26**	24	15	**22**	17	24.667

Table 8
Study comparison.

Methods	Heuristic	DRL	**This study**
Manual experience	Dependent	Less	Less
Rules	Fixed	Adaptive	Adaptive
Generalization	Poor	Good	**Better**
Space waste	High	Low	**Lower**
Perceive location	Limited	Intermediate	**Enhanced**

In order to solve the problem of poor performance of the current palletizing planning algorithm, a deep reinforcement learning online hybrid palletizing algorithm based on a mask attention mechanism is designed. According to the key point heuristic and the "space" heuristic, this paper proposes a new heuristic method – BP – and uses heuristics to update the palletizing state chain. Additionally, 3DPE is applied to the state chain, which enhances spatial encoding, leading to improved environmental comprehension and decision-making quality. Building on this, we designed a masked attention network to effectively encode the palletizing status information. By selectively masking attention weights between nodes, the network can focus more effectively on critical connections, thereby optimizing the palletizing performance. We have designed and implemented a 3D vision-based robotic system for online mixed palletizing. Experimental validation demonstrated that this system exhibits excellent practicality for online mixed palletizing tasks. Moreover, extensive experiments indicate that the proposed method achieves a certain improvement in final outcomes compared to other approaches.

For future research, we are particularly interested in applying our approach to unordered and irregularly shaped items, which presents a more challenging task. Additionally, in real-world palletizing processes, stability is not the only concern—other constraints such as weight distribution, priority, and weight limits often need to be considered. Therefore, future work could explore incorporating additional constraints to better align with practical application needs.

CRediT authorship contribution statement

Xingyu Mu: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation. **Quanmin Kan:** Writing – review & editing, Validation, Software, Conceptualization. **Yong Jiang:** Writing – review & editing, Methodology, Conceptualization. **Chao Chang:** Writing – review & editing. **Xincheng Tian:** Writing – review & editing, Supervision. **Lelai Zhou:** Supervision. **Yongguo Zhao:** Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.rcim.2024.102941.

Data availability

Data will be made available on request.

References

[1] Zhongyuan Lyu, Peng Lin, Daqiang Guo, George Q. Huang, Towards zero-warehousing smart manufacturing from zero-inventory just-in-time production, Robot. Comput.-Integr. Manuf. 64 (2020) 101932.

[2] Yi Zhang, Haihua Zhu, Dunbing Tang, Tong Zhou, Yong Gui, Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems, Robot. Comput.-Integr. Manuf. 78 (2022) 102412.

[3] Heng Xiong, Kai Ding, Wan Ding, Jian Peng, Jianfeng Xu, Towards reliable robot packing system based on deep reinforcement learning, Adv. Eng. Inform. 57 (2023) 102028.

[4] J.L. de Castro Silva, N.Y. Soma, N. Maculan, A greedy search for the three-dimensional bin packing problem: the packing static stability case, Int. Trans. Oper. Res. 10 (2) (2003) 141–153.

[5] Yongkui Liu, He Xu, Ding Liu, Lihui Wang, A digital twin-based sim-to-real transfer for deep reinforcement learning-enabled industrial robot grasping, Robot. Comput.-Integr. Manuf. 78 (2022) 102365.

[6] Hang Zhao, Yang Yu, Kai Xu, Learning efficient online 3D bin packing on packing configuration trees, in: International Conference on Learning Representations, 2021.

[7] Csaba Benedek, Andras Majdik, Balazs Nagy, Zoltan Rozsa, Tamas Sziranyi, Positioning and perception in LIDAR point clouds, Digit. Signal Process. 119 (2021) 103193.

[8] Hang Shang, Changying Liu, Ruijian Wang, Measurement methods of 3D shape of large-scale complex surfaces based on computer vision: A review, Measurement 197 (2022) 111302.

[9] Riccardo Monica, Jacopo Aleotti, Point cloud projective analysis for part-based grasp planning, IEEE Robot. Autom. Lett. 5 (3) (2020) 4695–4702.

[10] Hui Wang, Yu Huang, Guojun Zhang, Youmin Rong, A novel method for dense point cloud reconstruction and weld seam detection for tubesheet welding robot, Opt. Laser Technol. 163 (2023) 109346.

[11] Heng Xiong, Changrong Guo, Jian Peng, Kai Ding, Wenjie Chen, Xuchong Qiu, Long Bai, Jianfeng Xu, GOPT: Generalizable online 3D bin packing via transformer-based deep reinforcement learning, IEEE Robot. Autom. Lett. (2024).

[12] Paul E. Sweeney, Elizabeth Ridenour Paternoster, Cutting and packing problems: a categorized, application-orientated research bibliography, J. Oper. Res. Soc. 43 (7) (1992) 691–706.

[13] Sara Ali, António Galrão Ramos, Maria Antónia Carravilla, José Fernando Oliveira, On-line three-dimensional packing problems: A review of off-line and on-line solution approaches, Comput. Ind. Eng. 168 (2022) 108122.

[14] Alexandre Laterre, Yunguan Fu, Mohamed Khalil Jabri, Alain-Sam Cohen, David Kas, Karl Hajjar, Torbjorn S. Dahl, Amine Kerkeni, Karim Beguir, Ranked reward: Enabling self-play reinforcement learning for combinatorial optimization, 2018, arXiv preprint arXiv:1807.01672.

[15] Dongda Li, Zhaoquan Gu, Yuexuan Wang, Changwei Ren, Francis C.M. Lau, One model packs thousands of items with recurrent conditional query learning, Knowl.-Based Syst. 235 (2022) 107683.

[16] Wouter Kool, Herke Van Hoof, Max Welling, Attention, learn to solve routing problems!, 2018, arXiv preprint arXiv:1803.08475.

[17] Mhand Hifi, Imed Kacem, Stéphane Nègre, Lei Wu, A linear programming approach for the three-dimensional bin-packing problem, Electron. Notes Discrete Math. 36 (2010) 993–1000.

[18] Stefano Mutti, Giorgio Nicola, Manuel Beschi, Nicola Pedrocchi, L. Molinari Tosatti, Towards optimal task positioning in multi-robot cells, using nested meta-heuristic swarm algorithms, Robot. Comput.-Integr. Manuf. 71 (2021) 102131.

[19] Silvano Martello, David Pisinger, Daniele Vigo, The three-dimensional bin packing problem, Oper. Res. 48 (2) (2000) 256–267.

[20] Teodor Gabriel Crainic, Guido Perboli, Roberto Tadei, Extreme point-based heuristics for three-dimensional bin packing, Informs J. Comput. 20 (3) (2008) 368–384.

[21] Korhan Karabulut, Mustafa Murat 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.

[22] Chi Trung Ha, Trung Thanh Nguyen, Lam Thu Bui, Ran 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.

[23] Yong Wu, Wenkai Li, Mark Goh, Robert De Souza, Three-dimensional bin packing problem with variable bin height, European J. Oper. Res. 202 (2) (2010) 347–355.

[24] Ahmet Yarimcam, Shahriar Asta, Ender Özcan, Andrew J. Parkes, Heuristic generation via parameter tuning for online bin packing, in: 2014 IEEE Symposium on Evolving and Autonomous Learning Systems, EALS, IEEE, 2014, pp. 102–108.

[25] Ruizhen Hu, Juzhan Xu, Bin Chen, Minglun Gong, Hao Zhang, Hui Huang, TAP-net: Transport-and-pack using reinforcement learning, ACM Trans. Graph. (Proceedings SIGGRAPH ASIA 2020) 39 (6) (2020) 232:1 232:15.

[26] Fan Wang, Kris 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.

[27] Thanh-Hung Nguyen, Viet-Thang Tran, Phan-Quan Doan, Thi-Thoa Mac, A novel heuristic algorithm for online 3D bin packing, in: 2021 21st International Conference on Control, Automation and Systems, ICCAS, IEEE, 2021, pp. 1993–1997.

[28] Mingzhou Chen, Jiazhen Huo, Yongrui Duan, A hybrid biogeography-based optimization algorithm for three-dimensional bin size designing and packing problem, Comput. Ind. Eng. 180 (2023) 109239.

[29] Yu Huang, Daxin Liu, Zhenyu Liu, Ke Wang, Qide Wang, Jianrong Tan, A novel robotic grasping method for moving objects based on multi-agent deep reinforcement learning, Robot. Comput.-Integr. Manuf. 86 (2024) 102644.

[30] Haoyuan Hu, Xiaodong Zhang, Xiaowei Yan, Longfei Wang, Yinghui Xu, Solving a new 3d bin packing problem with deep reinforcement learning method, 2017, arXiv preprint arXiv:1708.05930.

[31] Jingwei Zhang, Bin Zi, Xiaoyu Ge Attend2pack, Bin packing through deep reinforcement learning with attention, 2021, ArXiv, arXiv:abs/2107.04333.

[32] Ruizhen Hu, Juzhan Xu, Bin Chen, Minglun Gong, Hao Zhang, Hui Huang, TAP-Net: transport-and-pack using reinforcement learning, ACM Trans. Graph. 39 (6) (2020) 1–15.

[33] Zifei Yang, Shuo Yang, Shuai Song, Wei Zhang, Ran Song, Jiyu Cheng, Yibin Li, Packerbot: Variable-sized product packing with heuristic deep reinforcement learning, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2021, pp. 5002–5008.

[34] Quanqing Que, Fang Yang, Defu Zhang, Solving 3D packing problem using transformer network and reinforcement learning, Expert Syst. Appl. 214 (2023) 119153.

[35] Shuo Yang, Shuai Song, Shilei Chu, Ran Song, Jiyu Cheng, Yibin Li, Wei Zhang, Heuristics integrated deep reinforcement learning for online 3d bin packing, IEEE Trans. Autom. Sci. Eng. 21 (1) (2023) 939–950.

[36] Shokhikha Amalana Murdivien, Jumyung Um, BoxStacker: Deep reinforcement learning for 3D bin packing problem in virtual environment of logistics systems, Sensors 23 (15) (2023) 6928.

[37] Richa Verma, Aniruddha Singhal, Harshad Khadilkar, Ansuma Basumatary, Siddharth Nayak, Harsh Vardhan Singh, Swagat Kumar, Rajesh Sinha, A generalized reinforcement learning algorithm for online 3d bin-packing, 2020, arXiv preprint arXiv:2007.00463.

[38] Hang Zhao, Chenyang Zhu, Xin Xu, Hui Huang, Kai Xu, Learning practically feasible policies for online 3D bin packing, Sci. China Inf. Sci. 65 (1) (2022) 112105.

[39] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, Michael Beetz, Towards 3D point cloud based object maps for household environments, Robot. Auton. Syst. 56 (11) (2008) 927–941.

[40] Martin A. Fischler, Robert C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.

[41] Zhang Yuankai, Jiang Yong, Tian Xincheng, Xu Xiaolong, Geng Yusen, Lai Min, A point cloud-based welding trajectory planning method for plane welds, Int. J. Adv. Manuf. Technol. 125 (3) (2023) 1645–1659.

[42] Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, Jimmy Ba, Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation, Adv. Neural Inf. Process. Syst. 30 (2017).

[43] Oriol Vinyals, Meire Fortunato, Navdeep Jaitly, Pointer networks, Adv. Neural Inf. Process. Syst. 28 (2015).

[44] Petar Velikovi, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, Yoshua Bengio, Graph attention networks, in: International Conference on Learning Representations, 2018.

[45] Yang Li, Tatsuya Harada, Lepard: Learning partial point cloud matching in rigid and deformable scenes, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 5554–5564.

[46] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, Yunfeng Liu, Roformer: Enhanced transformer with rotary position embedding, Neurocomputing (Feb.1) (2024) 568.

[47] Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, Samy Bengio, Neural combinatorial optimization with reinforcement learning, 2016, arXiv preprint arXiv:1611.09940.

[48] Hang Zhao, Qijin She, Chenyang Zhu, Yin Yang, Kai Xu, Online 3D bin packing with constrained deep reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 741–749.

[49] Roger Y. Tsai, Reimar K. Lenz, et al., A new technique for fully autonomous and efficient 3 d robotics hand/eye calibration, IEEE Trans. Robot. Autom. 5 (3) (1989) 345–358.

[50] Ju-Won Bae, Dong-Hoan Seo, Ju-Hyeon Seong, Two-dimensional camera and TOF sensor-based volume measurement system for automated object volume measuring, J. Adv. Mar. Eng. Technol. (JAMET) 47 (6) (2023) 419–426.

[51] Tao Peng, Zhijiang Zhang, Fansheng Chen, Dan Zeng, Dimension measurement and key point detection of boxes through laser-triangulation and deep learning-based techniques, Appl. Sci. 10 (1) (2019) 26.