European Journal of Operational Research 316 (2024) 584–596

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, manufacturing, transportation and logistics

3D dynamic heterogeneous robotic palletization problem

Wenbin Zhu [a], Ying Fu [a,b], You Zhou [a,c,*]

[a] School of Business Administration, South China University of Technology, Guangzhou, Guangdong, China
[b] Department of Industrial and Systems Engineering, University of Wisconsin–Madison, Madison, WI, USA
[c] Weldon School of Biomedical Engineering, Purdue University, West Lafayette, IN, USA

ARTICLE INFO

Keywords:
Packing
Online decision
Robotic palletization
Heterogeneous boxes

ABSTRACT

This paper proposes a practical robotic packing system to automate packing heterogeneous carton boxes into pallets, which is still handled manually in many distribution centers. The main challenge is to solve a 3D dynamic heterogeneous robotic palletization (DHRP) problem. Aside from the NP-completeness, there are two additional complexities in DHRP. First, the packing plan must be executable by a robotic arm without collision. Second, the packing decision must be made in real time with partial information. Tractable models are proposed to avoid collision for a common type of robotic arm and ensure the stability of the packing layout. An efficient algorithm is proposed to compute collision-free trajectories of the robotic arm during packing operations, which is embedded into a tree search algorithm to solve the semi-online counterpart of DHRP. Our semi-online algorithm is extended to solve the online version by adopting a Monte Carlo simulation strategy. A comprehensive set of test cases is generated based on realistic data to measure the performance of our algorithm. Numerical experiments show that our algorithm can produce feasible packing decisions in a few seconds for each incoming box on a modest personal computer, which is adequate for many practical setups. To help practitioners select the best setup for their application, numerical experiments are also conducted to analyze the impact of a few key design parameters in a robotic packing system on packing performance.

1. Introduction

The volume of logistic operations increased tremendously due to the widespread of e-commerce globally. A total of 63.5 billion parcels were delivered in 2019, representing a 25.3% annual increment in China alone, according to the State Post Bureau of China (StatePostBureau, 2020). To reduce logistic costs, conveyor belt systems, auto-guided vehicles, and robotic arms are widely used in large distribution centers to automate material handling processes such as order picking and packaging. However, packing heterogeneous carton boxes onto pallets remains a manual process in many centers due to the complexity of the underlying packing problem.

A robotic palletization system is proposed in this study to automate the palletization of heterogeneous carton boxes, as illustrated in Fig. 1(a). It consists of a conveyor belt system for transporting carton boxes, a measuring gate, a robotic arm, and an optional conveyor belt for transporting pallets. The dimensions of the N^k carton boxes that passed through the measuring gate become available for decision-making. The carton boxes within the robotic arm reach form a *loading buffer* of size N^b. The pallets within the robotic arm reach form a set of *open pallets* of size P. Each time, the robotic arm can grab one carton box from the loading buffer and load it onto an open pallet. When an open pallet is "full", it is *closed* and carried away by a forklift and replaced by an empty pallet.

At the end of a robotic arm is a gripper for handling cargo. Two types of grippers, splint robot gripper in Fig. 1(b) and vacuum gripper in Fig. 1(c), are commonly used for handling rectangular boxes. The former clamps a box from two opposite sides, and the latter sucks a box from one side. The vacuum gripper only contacts one face of a box, which makes it easier to avoid collision when the packing layout is complex. The metal panel of a vacuum gripper may be larger than the contact surface of a box, and the gripper may collide with existing boxes. The vacuum gripper works well with relatively light carton boxes, which account for the vast majority in a typical e-commerce warehouse. Therefore, this study focuses on developing a suitable collision avoidance model for the vacuum gripper.

The effectiveness of our proposed system relies on a novel model for the 3D dynamic heterogeneous robotic palletization (DHRP) problem that decides which box to load in each step and when an open pallet should be closed. In practice, logistic companies typically employ several standard-sized carton boxes. In this case, our problem is best classified as a Single Stock Size Cutting Stock Problem (SSSCSP) according to the typology by Wäscher et al. (2007). In applications where

* Corresponding author at: School of Business Administration, South China University of Technology, Guangzhou, Guangdong, China.
E-mail addresses: i@zhuwb.com (W. Zhu), ying.fu@wisc.edu (Y. Fu), zhou1129@purdue.edu (Y. Zhou).

https://doi.org/10.1016/j.ejor.2024.02.007
Received 12 August 2021; Accepted 7 February 2024
Available online 8 February 2024

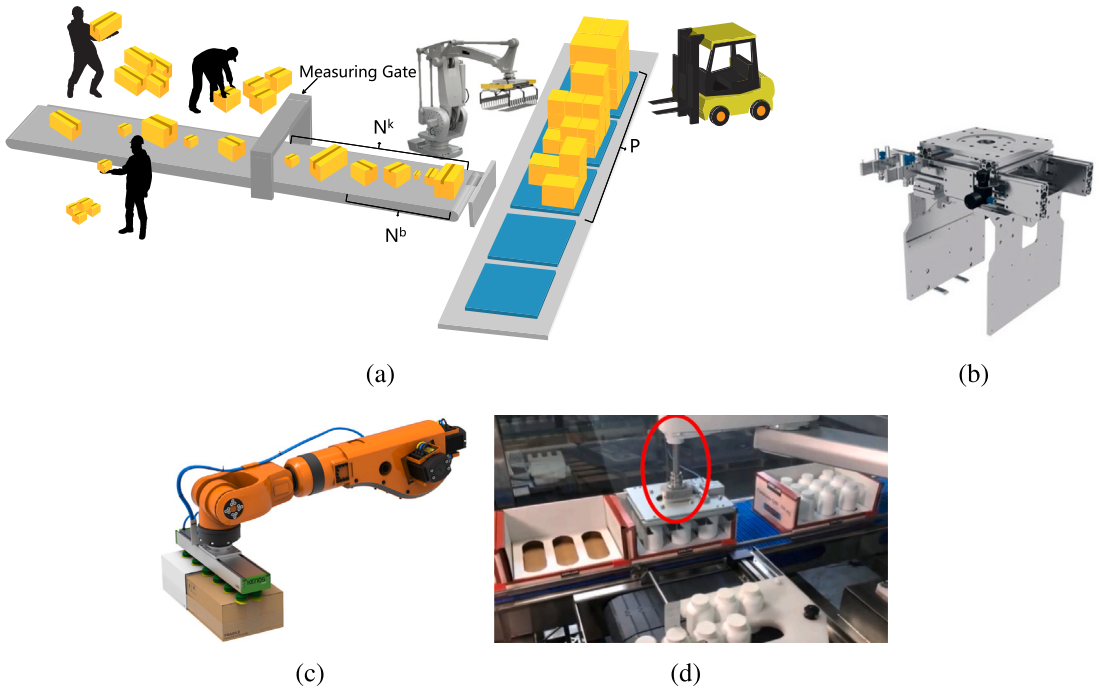

Fig. 1. (a) a robotic palletization system, boxes may be loaded onto the conveyor belt by workers or come from the automated upstream process; (b) splint robot gripper; (c) a vacuum gripper consists of 6 × 2 round suction cups mounted on a rectangular metal panel; (d) a vacuum gripper with 3 × 2 suction cups connected to robotic arm with an extendable rod (in red circle).

every box is distinct in dimensions, it is classified as a Single Bin Size Bin Packing Problem (SBSBPP). Both SSSCSP and SBSBPP are NP-hard.

In many application scenarios, the number of boxes to be loaded is large. In addition, the DHRP problem poses three new challenges: (1) the heterogeneity of carton boxes results in complex 3D layouts that necessitate careful planning to avoid collision between the robotic arm and the existing boxes; (2) the stability of the packing layout must be ensured at any stage to avoid collapse; (3) the dynamic nature of order arrival calls for an online decision with partial information.

Here are our main contributions,

- Proposed a tractable collision avoidance model and stability model. Both models are compatible with the Integer Linear Programming packing model by Chen et al. (1995) so researchers can extend ILP models to robotic packing problems. Feasible solutions to both models can be efficiently computed so that fast heuristic algorithms can be developed to handle large instances encountered in practice.
- Developed an efficient heuristic algorithm for the DHRP problem and demonstrated its performance through extensive numerical experiments.
- Generated test instances with realistic characteristics for typical scenarios and analyzed how critical parameters in the robotic packing system and our heuristic algorithms impact the overall packing performance.
- Implemented our algorithm available online, enabling solution developers and practitioners to follow our procedure to decide their scenario's optimal setup.

The rest of this paper is organized as follows. Section 1.1 summarizes related literature. Section 2 formally defines the DHRP problem. Section 3 presents our collision avoidance model for the robotic arm and introduces a new formulation to ensure the stability of the packing layout. Section 4 develops an efficient procedure for computing the collision-free trajectory of the robotic arm, which is embedded into a tree search algorithm that employs the Monte Carlo simulation strategy to solve the DHRP problem. Section 5 conducts numeric experiments to validate the effectiveness of our proposed algorithms and to analyze the impact of key system parameters on the overall performance. Section 6 concludes our work and points out further research directions.

1.1. Related literature

Robotic arms are first employed to pack pallets in manufacturers' production lines, where every pallet only contains a single type of carton boxes (Bischoff et al., 1995; Kocjan & Holmström, 2008; Terno et al., 2000). In this scenario, boxes are packed layer by layer from the bottom up, and all boxes in the same layer are arranged at the same height. Such a packing layout can be easily executed by a robotic arm. Avoiding potential collision is trivial and, therefore, not explicitly discussed. Martello et al. (2007) introduced the concept of a robotic packable layout in an offline setting. A packing layout is assumed to be executable by a robotic arm if there exists a packing order so that every box is on the right, in the front, or above all preceding boxes. This model omits the physical dimensions of specific robotic arms and their trajectories during packing. Schuster et al. (2010) studied packing by robotic arms equipped with a vacuum gripper, where every box is assumed to be moved to its final position by the vertical move. The potential collision between the robotic arm and existing boxes is not explicitly modeled. Agarwal et al. (2021) decouple the computation of optimal packing layout from its physical execution by a robotic arm. They mainly focus on mitigating the imperfection of the robotic arm. Their fault recovery module (FRM) will readjust the placement of a box if sensor feedback indicates a large deviation between the actual location and the planned position. FRM calls for human intervention when a robotic arm cannot execute an action. They developed a simple iterative heuristic for the offline 3D-BPP to find packing layouts without considering stability constraints or the impact of a robotic arm. Wang and Hauser (2019) proposed a sequential heuristic to pack 3D irregular

objects into a single container by a robotic arm. The motion of the robotic arm is restricted to vertical only, which is typical for loading items into a carton box from the top. Their method is tailored to address the complex stability constraints and manipulation feasibility caused by the non-convex shape of objects. Wang and Hauser (2021) further extends this work to handle online settings where items are known and arrive in a non-deterministic order. There is a stream of literature on palletizing mixed cases in the robotic field; see the book chapter by Agha et al. (2000) for a good introduction. The focus is usually on improving the operation efficiency of hardware, such as real-time control of robotic arms. Packing decisions are typically made by simple heuristics (Demisse et al., 2012; Nguyen-Vinh et al., 2022).

Bortfeldt and Wäscher (2013) reviewed many practical constraints in container loading, among which load stability is an important constraint widely studied by many authors. Two types of stability exist: static (or vertical) stability during loading operation and dynamic (or horizontal) stability during transportation. Most studies focus on static stability, which is the most relevant to this study. Many approaches have been devised to achieve static stability. Galrão Ramos et al. (2016) broadly classify these approaches into three categories: **full base support**, **partial base support** and **static mechanical equilibrium**.

In full base support approaches, such as Wang et al. (2013), Zhu and Lim (2012), the entire bottom face of every box is in contact with the top face of other boxes or the base of the pallet. Full base support ensures static mechanical equilibrium when all items are rectangular, but it is too restrictive and hinders space utilization. Many studies, such as Mack et al. (2004), relax it to partial base support to improve space utilization, where at least α percent of the bottom area of every box is supported from below. This model is inadequate for robotic packing; see Section 3.1.1 for details.

Static mechanical equilibrium approaches, such as de Azevedo Oliveira et al. (2021), De Castro Silva et al. (2003), Galrão Ramos et al. (2016), Wang and Hauser (2019), Whiting et al. (2009) require force and torque balance for each box in a layout. A SME model includes six linear equations for each box and 12 quadratic constraints for every pair of contacting boxes. Such an approach is intractable; see Section 3.1.2 for details. In addition, the rigid body assumption is inappropriate for corrugated cardboard carton boxes since they may deform substantially under load. The finite element method is typically required to model force distribution; see Jiménez-Caballero et al. (2009). However, FEM models need much information, such as Young's modulus of every corrugated cardboard's exact distribution of weights inside each box, which is hard to obtain in packing applications.

Ali et al. (2022) provides a comprehensive review of online 3D packing problems. Classical online setup strictly forbids access to future information, i.e., $N^k = 1$. All solution approaches are doomed myopic — the packing decision for the current box is solely determined by the fitness of the box and a candidate position in the current layout without considering the impact of future boxes. The differences mainly lie in whether the fitness function is handcrafted (Ha et al., 2017) or trained via deep reinforcement learning (Verma et al., 2020; Zhao et al., 2021). We show that a little future knowledge combined with a tree search can significantly improve decision quality since future knowledge can be acquired with very little cost. Our dynamic setup with $N^k > 1$ is a meaningful extension to the classical online setup. Classical online setup also forbids reordering of the loading sequence, i.e., $N^b = 1$, which may hinder the efficient utilization of loading spaces. We relaxed this restriction in our dynamic setup. Saito and Asari (2019) studied two special cases of the dynamic setup where $N^k = N, N^b = 1$ and $N^k = N^b > 1$ and called them semi-online.

Among the state-of-the-art offline 3D packing algorithms (Alonso et al., 2020; Elhedhli et al., 2019; Fanslau & Bortfeldt, 2010; He & Huang, 2011; Zhu et al., 2021; Zhu & Lim, 2012; Zudio et al., 2018), a common strategy to reduce the complexity of the problem is to group boxes into blocks (including layers and walls) and load a block of boxes in each step. In the online/dynamic setting, this strategy may fail, as loading algorithms' arrival order of boxes is not controllable. Boxes in the desired block may not be available in the first few arrived boxes. In addition, it is hard to generate blocks where a robotic arm can pack every box without collision. Another common strategy adopted by Parreño et al. (2010, 2008) to improve space utilization is to represent the free space in pallets by a set of overlapping cuboids, called *maximal space representation*. This strategy is compatible with our problem settings and adopted in this study. Zhu et al. (2012) has shown that a simple two-step look ahead tree search with an appropriate fitness measure can outperform complex search strategies. This search strategy is also adopted in this study.

2. Problem definition

The 3D dynamic heterogeneous robotic palletization (DHRP) problem is defined with a reference to Fig. 1(a). A total of N rectangular boxes arrive at the measuring gate sequentially on a conveyor belt, where the dimension of the ith box is $L_i \times W_i \times H_i$. A box first passes through a measuring gate where the exact dimensions become available and then arrives at the robotic arm. For simplicity, we assume the following. There are always N^k boxes between the measuring gate and the robotic arm, forming the *waiting area*. The first N^b boxes in the waiting area are within the operation range of the robotic arm and are called the *loading buffer*. The speed of the conveyor belt can be dynamically adjusted to match the processing speed of the robotic arm so that the exact positions of the boxes on the conveyor belt become irrelevant, and only the arrival order will affect the packing decision.

The robotic arm can grip exactly one box from the loading buffer in each step and load it onto a *target pallet* selected among P *open pallets* within its operation range. The dimension of a pallet is given by $L \times W \times H$, where H represents the maximum loading height. When an open pallet is too "full" to accommodate more boxes, it is *closed* and replaced by an empty pallet. When P is large, we assume a proper conveyor belt is utilized to move the open pallets back and forth at an appropriate speed to ensure the target pallet is within the operating range of the robotic arm. When P is small, say $P = 1$, no conveyor belt is needed. With this assumption, the exact positions of the open pallets become irrelevant.

The goal is to load all N boxes into a minimum number of pallets (equivalent to maximizing average volume utilization overall used pallets) while respecting the following common packing constraints:

- *axis-aligned*: all boxes are placed with sides parallel to the sides of pallets.
- *containment*: all boxes in a pallet must be fully contained within the boundaries of the pallet.
- *nonoverlapping*: any pair of boxes inside a pallet are interior-disjoint.
- *orientation-limit*: although a 3D box has six possible rotations, some rotations are not permitted for some boxes due to safety/stability considerations.

The following constraints to ensure packing can be carried out by a robotic arm,

- *stepwise stable support*: every box must be adequately supported so that the carton boxes in the pallet will not collapse during the entire process of packing. Stability is required at every step, not only in the final configuration.
- *irrevocable packing*: once a box is loaded into a pallet, it cannot be moved. Rearrange existing boxes in a pallet is not allowed. It increases the complexity of packing decisions under online settings; see Fig. 2.
- *stable gripping*: a box can be tightly gripped during packing by a robotic arm equipped with a vacuum gripper connected to the robotic arm with an extendable rod, see Fig. 1(d).
- *collision-free*: there exists a collision-free motion path for the robotic arm to pack every box.

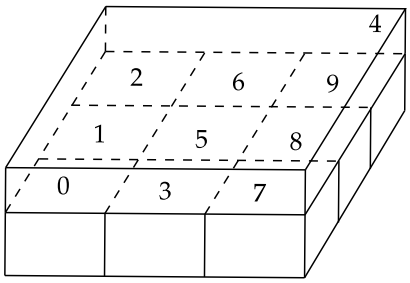

Fig. 2. Arrival sequence matters. In offline settings, nine smaller boxes can be nicely arranged to occupy the entire pallet, with the large box 4 packed on top. In the online setting with irrevocable packing, box 5 cannot be pushed to the desired position by a large gripper when boxes arrive in order. Box 5 is forced to be packed on top of box 4.

3. Modeling

3.1. Stepwise stable support

3.1.1. The drawback of the partial base support model

In the partial base support model, a box is considered stable if at least α percent of its bottom area is supported from below. This is not an accurate model for stability. On the one hand, we may have a stable configuration where the support area is nearly zero; see Fig. 3(a). On the other hand, we may still have an unstable configuration with a support area as large as 50%; see Fig. 3(b). When packing is executed manually, such inaccuracy is acceptable as human hands are very flexible, and workers can easily hold temporarily unstable boxes or fill the gaps with filling materials. In contrast, a robotic arm requires that all boxes are stably supported in every loading step. Therefore, partial base support is inadequate.

3.1.2. The drawback of the static mechanical equilibrium model

In the static mechanical equilibrium model, a layout is stable if force and torque are balanced for each box and the forces between two boxes are compressive (boxes are not glued together; hence, there is no tensile force). According to Whiting et al. (2009), for rigid bodies, the force can be positioned at every vertex of the contact surface to model a linear force distribution across the surface, see Fig. 4(a).

Adapted the SME model by Wang and Hauser (2019), the boxes are in static equilibrium if we can find a set of forces satisfying the following conditions:

Force balance: for all box i,

$$\sum_{C\in\mathcal{C},i=C^+} f_C + \sum_{C\in\mathcal{C},i=C^-} -f_C + m_i g(-\mathbf{n}) = 0 \tag{1}$$

Where $\mathcal{C}$ is the set of all contact points, every contact point C involves a pair of boxes denoted by C^+ and C^- with box C^+ receiving a force f_C and C^- receiving an opposite force $-f_C$, cm_i is the center of mass of box i, and $\mathbf{n}$ is upward vertical direction.

Torque balance: for all box i,

$$\sum_{C\in\mathcal{C},i=C^+} (\mathrm{cm}_i - C)\times f_C + \sum_{C\in\mathcal{C},i=C^-} (\mathrm{cm}_i - C)\times(-f_C) = 0 \tag{2}$$

Compressive force only: for all contact point C,

$$f_C \cdot \mathbf{n}_C \geq 0 \tag{3}$$

where $\mathbf{n}_C$ is the normal at contact point C.

Coulomb's law of friction: for all contact point C,

$$\|f_C^{\perp}\| \geq \mu_C(f_C \cdot \mathbf{n}_C) \tag{4}$$

where $f_C^{\perp} = f_C - (f_C \cdot \mathbf{n}_C)\mathbf{n}_C$ is the frictional force and μ_C is friction coefficient. Since Coulomb's law is a quadratic constraint, incorporating it into a packing model will turn an integer linear program into a quadratically constrained quadratic problem (QCQP) with integer variables, which is intractable for a large number of boxes. Note that (1) and (2) each include three linear equations for every box. Each pair of contacting boxes has a contact area with four vertices, and each vertex contributes three quadratic inequalities to (4).

Model (1)–(4) is a good approximation of stability only for boxes made of hard materials such as wood or metal, where the deformation of boxes is small and can be safely ignored. For commonly used corrugated cardboard carton boxes, the deformation can be substantial since it changes the shape of a box and the positions of boxes above it, which will, in turn, change force distribution among boxes. An accurate model must take deformation into account. Although finite element method (FEM) (Jiménez-Caballero et al., 2009) has been proposed to model deformation and force distribution inside corrugated cardboard, they are too complex to be incorporated into a packing model. In addition to complexity, FEM models require much information, such as Young's modulus of every corrugated cardboard and the exact distribution of weights inside each box, which is hard to obtain in packing applications. Hence, we abandoned the static mechanical equilibrium approach and developed a more straightforward and practical heuristic in this study.

Note that the SME model cannot be computed incrementally. The force distribution between all existing boxes will change after loading a new box. To decide the best position to place a new box, a SME model must be solved for each candidate position of the new box.

3.1.3. The heuristic rule for stable support

The following heuristic rule is adopted to ensure the stability of a box. The bottom face of a box is divided into 2×2 equal regions A, B, C, D, see Fig. 5. The box is stable if at least $\Gamma = 3$ regions are supported from below by any other box or the base of the pallet.

A region is supported from below by box j if its H-coordinate coincides with the top face of the box j and it overlaps with the top face in both L- and W-axis. We require the overlap in both L- and W-axis to exceed α times the length and width of the box i so that the support area is large enough to ensure stability.

In most cases, $\Gamma = 3$ is sufficient to ensure a stable packing layout. However, in application scenarios where box sizes vary, $\Gamma = 3$ may be insufficient. Fig. 6 illustrates an unstable packing layout that satisfies all stable support constraints for $\Gamma = 3$. Since the center of geometry of the top box, i is outside the convex hull of the contact surfaces $ABCDE$, it will fall due to gravity. The question is how likely this type of instability will occur in practice. According to our computational results in Section 5.6, among 288 packing instances, 163,200 boxes are loaded, and no box's center of geometry lies outside the convex hull. In the worst case, if it does occur, we may increase α or set $\Gamma = 4$ to ensure stability.

In fact, our more straightforward rule agrees with SME model (1)–(4) with very high probability. According to our results in Section 5.6, among 163,200 boxes loaded, only 2,256 boxes are unstable by the SME model. The disagreement is less than 1.4%. Hence, it is better to employ our more straightforward heuristic rule instead of the SME model in our optimization model when searching for the best loading decisions. Invoke the SME model after the best decision is found and before the robotic arm executes it. If the SME check fails, remove the decision from the candidate pool and reoptimize to find an alternative.

3.2. Stable gripping

For stable gripping constraints, we made the following assumption about the operating characteristics of the robotic arm. The vacuum gripper is a rectangle of dimension $L^g \times W^g$, with a grid of $M^g \times N^g$ circular vacuum suckers evenly distributed inside the gripper. The number of suckers and their diameters varies. A gripper with more and larger suckers can lift heavier boxes. We assume the edges of a gripper are always parallel to the edges of boxes when gripping a box. A

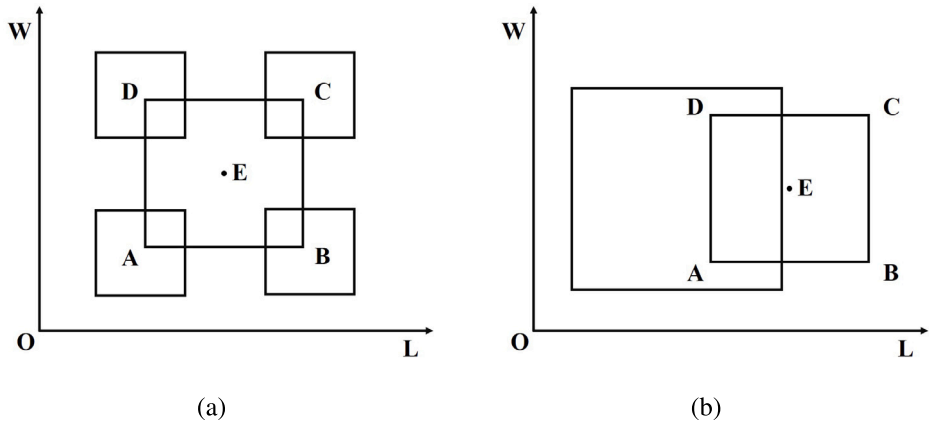

Fig. 3. (a) a tiny supporting area can ensure the stability of the top box, where point E and rectangle $ABCD$ are the centroid and the base of the top box, respectively; (b) nearly half of the base area is supported, yet the top box is unstable.

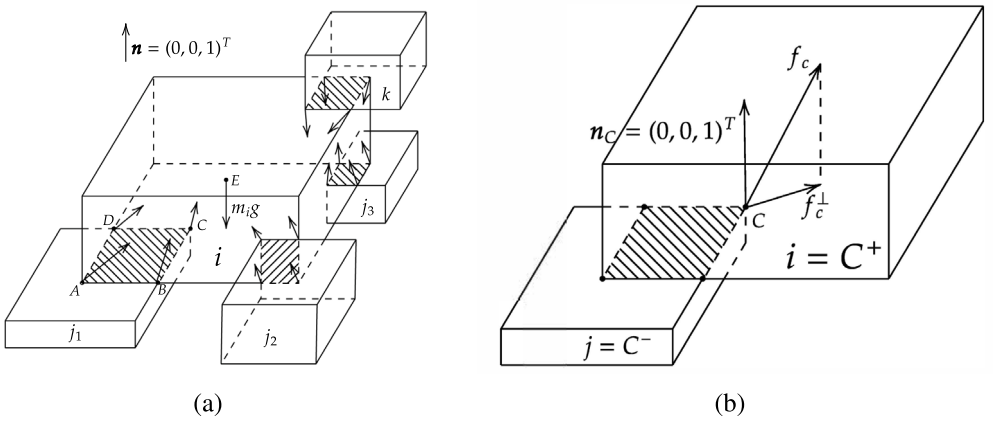

Fig. 4. (a) Box i is supported from below by box j_1 and j_3, supports box k, and is adjacent with box j_2. The contact forces applied to box i are distributed linearly across the contact surface, modeled by forces placed at each vertex. Gravity is applied at the center of mass E. (b) A pair of box i and j contact at point C are denoted as C^+ and C^-, respectively. By convention, the force applied to C^+ is f_C, and its negative is applied to C^-. $\mathbf{n}_C$ is the normal at contact point C.

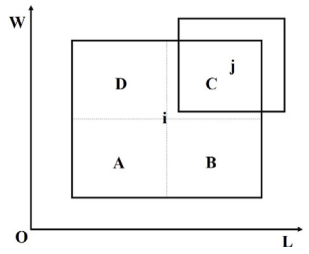

Fig. 5. Bottom face of box i is divided into region A, B, C, D. Region C is supported from below by box j.

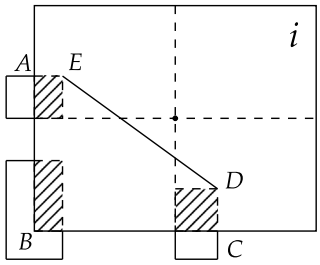

Fig. 6. An unstable packing layout satisfies all stable support constraints when $\Gamma = 3$.

vacuum sucker will function only if it is fully inside the contact surface. The relative position of a gripper and the contact surface determines the number of functioning suckers. Fig. 7 illustrates six functioning suckers. In this study, we simplify the problem by ignoring the weight of carton boxes. Given a contact surface, we always choose a relative position to maximize the number of functioning suckers so that the gripper can provide maximum suction. While maintaining a maximum number of functioning suckers, we allow the relative position to be adjusted to increase packing flexibility, as illustrated in Fig. 7.

3.3. Collision avoidance

We assume the target pallet is placed at a proper location within the operating range of the robotic arm. Both the pallet and robotic arm are in the first quadrant; see Fig. 8(a) .

When loading small boxes, the panel of the vacuum gripper may be larger than the contact face. The robotic arm, its gripper, or the box to be loaded may collide with existing boxes in the pallet. Finding a collision-free path is known as motion planning, which is a challenging problem when packing layouts are complex. We break it into two simpler planning stages; see Fig. 8(b). In the first stage, the robotic arm transfers box i outside the pallet via a curved path and rotates the gripper to align the edges of the box i with an axis. In the second stage, it extends the rod attached to the gripper (Fig. 1(d)) to push box i straightly along L-axis without any rotation. We call it L-push. The first stage can be simplified by treating the entire loading space as

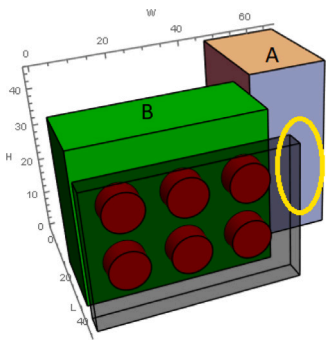

Fig. 7. The right side of the gripper (yellow circle) will collide with box A during inward pushing (along the L-axis). Slightly move it towards the left while keeping box B still can avoid the collision.

one rectangular obstacle. Commercial motion planners such as OMPL[1] can easily compute a collision-free path. Hence, this study only focuses on motion planning in the second stage. This L-axis push creates two rectangular trajectories labeled A (light purple) and B (orange). A and B represent the respective trajectories of the gripper and the box during their horizontal movement. There is a collision-free path if and only if rectangles A and B are interior-disjoint with existing boxes. Since A and B are axis-aligned, they can be handled the same way as carton boxes. An exciting consequence is that our collision avoidance model can be easily formulated as liner constraints in an Integer Linear Programming model for packing problems, such as Chen et al. (1995), opening the possibility of extending existing exact packing algorithms to robotic packing problems.

4. Algorithm for 3D online robotic palletization problem

4.1. Online decision via simulation

The main challenge in an online setting is that we only have limited knowledge when packing the current box. We only know the first N^k boxes in the waiting area and do not know what will come in the future. A good packing position based on current knowledge may turn out to be a poor choice when future knowledge becomes available. Ideally, we could employ a probabilistic model to describe the probability of all possible futures compatible with current knowledge. That is to enumerate all possible box arrival sequences whose first N^k boxes match those in the waiting area and assign a probability to each. We try to find the best decision for each possible sequence and then pick the decision to maximize the expected volume utilization among all possible sequences. Maximizing the expectation is impractical since there are $(N - N^k)!$ possible sequence. A common heuristic is to apply Monte Carlo simulation to generate a few sequences to form a sample, then find a decision that maximizes the sample average instead of expectation.

To evaluate the effectiveness of our approach. We implemented a simulator GreedyBySimulation that simulates the arrival of boxes from the conveyor belt, packs boxes according to the decision made by our main algorithm GLAS, and tracks the total number of pallets used.

The while-loop on line 2 in GreedyBySimulation simulates the boxes' arrival from the conveyor belt, one at a time. It invokes packing decision maker GLAS on line 5. GLAS can only use knowledge about boxes in the waiting area $\mathcal{W}$, loading buffer $\mathcal{B}$ and the packing layout of open pallets $\mathcal{O}$ to return a packing decision p, indicating which box from the loading area $\mathcal{B}$ should be packed into which open pallet at what location. It also includes the box's orientation, contact face, and the pushing axis. GreedyBySimulation packs the box according to p, update packing layout of involved open pallet accordingly on line 7. If all open pallets are too full to accommodate any box in the loading buffer, no packing is possible, as indicated by $p = NULL$. In this case, ReplaceFullPalletByEmpty is called to replace the fullest open pallet with an empty one in line 8.

Our main algorithm GLAS implements the Monte Carlo strategy. It generates a set of box arrival sequences compatible with boxes in the waiting area in line 2. For each fixed arrival sequence, the problem becomes semi-online and is solved by our tree search algorithm GLA in line 4. GLA returns the best packing decision for the first box. We vote for the best packing decision in line 5 and return the decision with the most votes in line 6.

In line 1 of algorithm GLAS, all validate placements are generated. A valid placement p places a box from $\mathcal{B}$ at a location inside one of the open pallets $\mathcal{O}$, and the robotic arm can execute it without collision. In line 2 of algorithm GLAS, a set of box arrival sequences are generated to simulate typical application scenarios. The generated sequences always start with $\mathcal{W}$. An application scenario describes how many box types are utilized and their relative frequencies computed from historical data. A sequence is generated by randomly picking one box at a time from available box types with probability proportional to their relative frequencies. The generation process stops once the total volume of the boxes fills up the remaining spaces of all current open pallets.

4.2. Semi-online decision via incomplete tree search

Once the entire loading sequence s is known, the DHRP problem becomes semi-online – to load all boxes in s into the set of open pallets $\mathcal{O}$ to maximize the space utilization. We adopt the greedy lookahead algorithm (GLA) by Zhu and Lim (2012) to solve this semi-online problem.

Given an arrival sequence s, GLA attempts to find the best solution by incomplete tree search. The search depth is limited to d, and only the top k fittest children are explored to prevent the exponential explosion of the search space; see Fig. 9(a). The fitness of a placement is defined by Eq. (9) in Section 4.4. For a node at depth $d = 2$, a greedy strategy is invoked to load one box at a time until all remaining boxes in s are loaded, or all open pallets are full to obtain a final state. During the greedy phase, the fittest placement is chosen at each step. The final state with the largest volume utilization is considered the best final state. Volume utilization is defined by the total volume of boxes in the open pallets divided by the number of open pallets employed. The path to the best final state is highlighted in thick lines in Fig. 9(a). The first placement decision on the best path is returned by GLA as the best placement for the root node.

A node η in the search tree includes a sequence $\eta.s$ of boxes to be loaded, in which the first N^b boxes are reachable by the robotic arm. A node η also records a list of open pallets $\eta.openPallets$. The state of an open pallet o includes a list of rectangular cuboids $o.placedBoxes$, each representing a placed box, and a list of rectangular cuboids $o.freeSpaces$, each representing a free space. The maximal space concept by Parreño et al. (2008) is employed to represent free space in open pallets. Initially, the entire loading space of an open pallet is represented by a rectangular cuboid. After a box is loaded, all free spaces that overlap with the box are split into smaller rectangular cuboids as follows. When a box overlaps with a free space, the residual space of the free space can be covered by at most six cuboids, each corresponding to one face of the box; see Fig. 9(b) for the cuboid corresponding to the top face.

[1] https://ompl.kavrakilab.org/

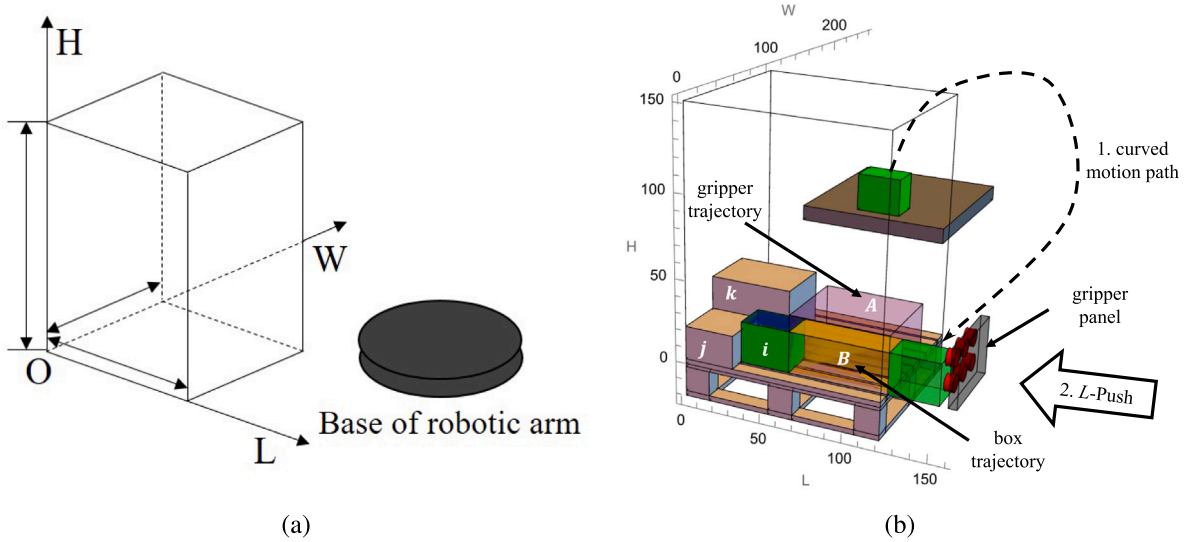

Fig. 8. (a) Local coordinate system, where the base of the robotic arm and the target pallet (3D box) are in the first octant; (b) The robotic arm transfers box i along a curved path outside the loading space and then pushes it along the L-axis. This L-axis push creates two rectangular trajectories labeled A (light purple) and B (orange). A and B represent the respective trajectories of the gripper and the box during their horizontal movement. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Algorithm 1 Place one box at a time by Greedy via Monte Carlo simulation

GreedyBySimulation(s)

$\mathcal{O}$ = open pallets
while $s \neq \emptyset$
    $\mathcal{B}$ = first N^b boxes in s
    $\mathcal{W}$ = first N^k boxes in s
    // find the best placement by Greedy Look-Ahead via Simulation
    p = GLAS($\mathcal{B}, \mathcal{W}, \mathcal{O}$)
    if $p \neq NULL$
        load according to p and remove loaded box from s
    else $\mathcal{O}$ = ReplaceFullPalletByEmpty($\mathcal{O}$)

GLAS($\mathcal{B}, \mathcal{W}, \mathcal{O}$)

$\mathcal{P}$ = ValidPlacement($\mathcal{B}, \mathcal{O}$)
$\mathcal{S}$ = generate sequences starting with $\mathcal{W}$
for each $s \in \mathcal{S}$
    // solve the semi-online problem corresponding to s
    // find the best placement p for the current step
    p = GLA($s, \mathcal{O}, \mathcal{P}$)
    increase $BestCount$ of p by 1
return p with largest $BestCount$ among $\mathcal{P}$

4.3. Enumerating feasible placements for a node

For a node η, all feasible placements can be enumerated as follows. For each box i in the first N^b boxes in $\eta.s$ and each permitted orientation $ort \in \mathcal{R}_i$ of the box, every free space r in every open pallet that is large enough to accommodate the box form a triplet (i, ort, r). A triplet gives rise to many feasible placements, one for each location loc where it can be stably supported. Assume box i is pushed to loc by W-push. A candidate placement (i, ort, r, loc) is feasible if,

(1) The trajectory B of box i during W-push will not overlap with existing boxes;
(2) An appropriate orientation and displacement of the gripper can be found so that the box can be firmly grabbed and the trajectory A of the gripper will not collide with existing boxes.

If any of above condition fails, box i cannot be pushed to loc by W-push.

Recall that the location of box i is identified by its vertex closest to the origin. When box i is pushed to $loc = (\underline{l}_i, \underline{w}_i, h_i)$ along W-axis, a box j in the pallet may block box i only if,

$$\underline{w}_j + \Delta w_j > \underline{w}_i \tag{5}$$

where $\underline{w}_j$ and Δw_j are the location and length of box j along W-axis, respectively. When inequality (5) holds, box j is called a *potential blocker*, its projection onto W-axis overlaps with the projection of the trajectory B of box i, see Fig. 10(a).

Location loc can be enumerated as follows. The location of a free space r is identified by its vertex closest to the origin, $(r.l, r.w, r.h)$. $loc.w$ is always set to $r.w$; that is, we will push the box as close to the origin

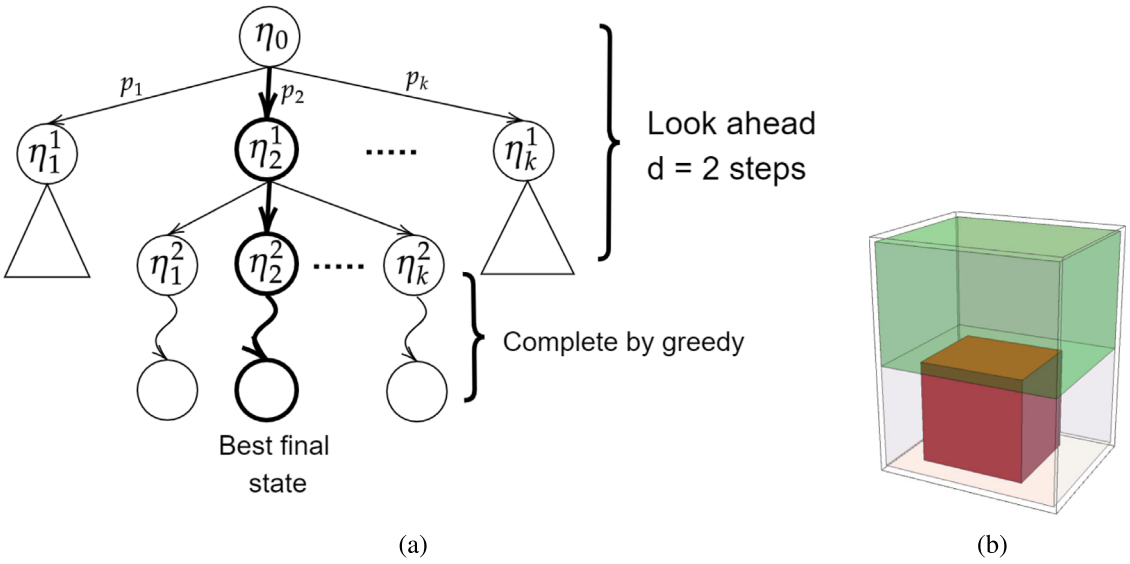

Fig. 9. (a) GLA with search depth $d = 2$, the path to the best final state is highlighted with thick lines, p_2 is the best placement. (b) A box (red) inside a free space, its top face defines a residual space (green).

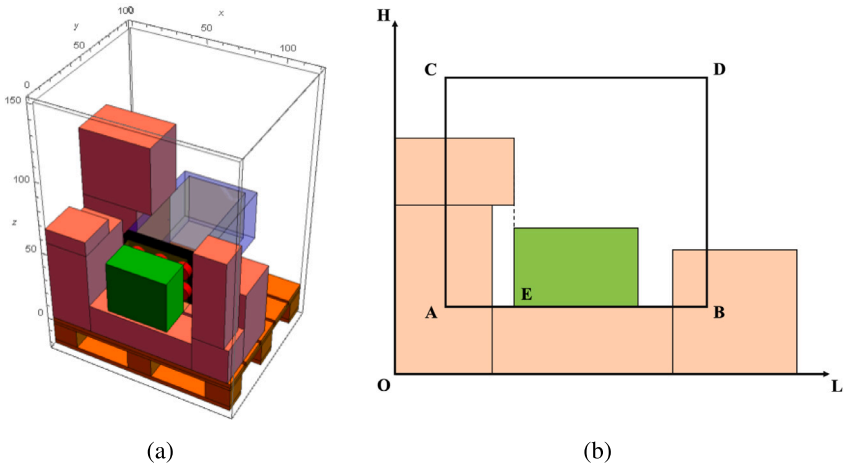

Fig. 10. (a) A W-push example. (b) $ABCD$ is the projection of a free space onto LH plane and pink rectangles are projections of potential blockers, E is the final position of box i. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

as possible without leaving the free space. All potential blockers are identified by inequality (5) with $\underline{w}_i = r.w$ and their projection onto LH plane are pink rectangles in Fig. 10(b). The projection of free space r is the rectangle $ABCD$. If the projection of box i (green rectangle) can be placed inside $ABCD$ without overlapping with any potential blockers, box i can be pushed along W-axis without collision. Finding all feasible positions is reduced to the 2D box placement problem, which can be solved in $O(n \log n)$ time by the algorithm of Zhu et al. (2016), where n is the number of blockers. Point E in Fig. 10(b) is one feasible position. Since slightly moving the green box towards the right is also a feasible position, there are infinitely many feasible positions in this example. We will restrict ourselves to the feasible positions where the faces of the new box i coincide with the faces of some placed boxes or free space. More precisely, $loc.l$ and $loc.h$ are from the following sets $\mathcal{L}$ and $\mathcal{H}$, respectively,

$$\mathcal{L} = \{0\} \cup \{\underline{l}_j | j \in \mathcal{B}\} \cup \{L - \Delta l_i\} \cup \{\underline{l}_j + \Delta l_j - \Delta l_i | j \in \mathcal{B}\} \quad (6)$$

$$\mathcal{H} = \{0\} \cup \{\underline{h}_j | j \in \mathcal{B}\} \cup \{H - \Delta h_i\} \cup \{\underline{h}_j + \Delta h_j - \Delta h_i | j \in \mathcal{B}\} \quad (7)$$

$$\mathcal{B} = \text{index set of boxes in the same pallet as } i \quad (8)$$

Next, we proceed to check whether box i at location loc is stable and discard the location if not stable. Box i is stable if at least three out of its four bottom areas are supported by some boxes from below, which can be checked by enumerating all existing boxes.

Finally, a location loc is feasible only if a gripper orientation exists so that it can send the box to loc without colliding with any existing boxes. We will try both possible orientations; either the width or the length of the gripper is placed horizontally (aligns with L-axis). Although there may be infinitely many possible displacements between the gripper and the surface of the box i it grips, we only try one extreme displacement — we will place the gripper as far away as possible from the origin while ensuring stable gripping of the new box. Since our algorithm tends to concentrate loaded boxes towards the origin, such choices increase the chance of avoiding the collision of the gripper and the existing boxes.

Two feasible placements with the same box i, orientation ort, and final location loc are equivalent, even if they are from different free spaces (in the same pallet) or different pushing axes. Because they have the same effect on future loading decisions, when enumerating feasible placements, we will only keep feasible positions with distinct (i, ort, loc).

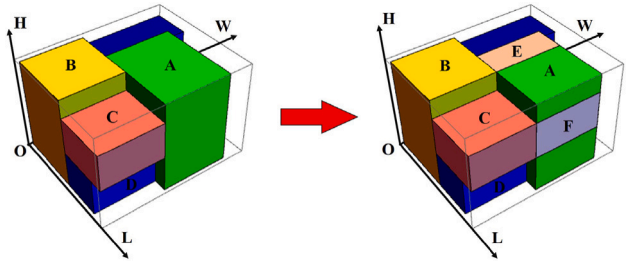

Fig. 11. Computation of regularity.

4.4. Local fitness measure

Given a feasible placement p, representing placing box $p.i$ with an orientation $p.ort$ at the location $p.loc$, its fitness is measured based on: (1) regularity; (2) contact area; (3) support ratio, and (4) height of final position as follows,

$$f(p) = (\text{Regularity}(p), \text{ContactArea}(p), \text{SupportRatio}(p), -loc.h) \tag{9}$$

A placement p_1 is considered better than p_2 if $f(p_1)$ is lexicographically greater than $f(p_2)$.

Regularity measures how well the continuity of free space in the pallet is preserved after the placement. It is well known that continuous large free spaces are easier to fill up densely than scattered small free spaces. Therefore, avoiding breaking free spaces into fragments helps increase space utilization. Our regularity measure is inspired by the caving degree by Huang and He (2009) and the skyline concept by Wei et al. (2011).

Regularity is computed as follows. A box j is *adjacent* to box i along W-axis, if $\underline{w}_j + \Delta w_j = \underline{w}_i$. In Fig. 11, boxes B, C, and D are all adjacent to box A along W-axis, while box A is not adjacent to B, C, or D. Adjacency along the L- and H-axis are similarly defined. A box j is *aligned* with box i along H-axis, if $\underline{h}_j + \Delta h_j = \underline{h}_i + \Delta h_i$. Alignment along the L- and W-axis are similarly defined. In Fig. 11, box B aligns with box A along H-axis, and box C aligns with box A along L-axis.

A box such as B in Fig. 11 adjacent to current box A along the W-axis and aligned with A on the H-axis may overlap with A along the L-axis. Such overlap contributes to a *regular area*, rectangle E, on the H-face (the face perpendicular to H-axis with a larger H-coordinate) of the current box A.

The total regular area contributed by all existing boxes on the H-face of the current box is defined to be its H-regularity. To understand the intuition behind H-regularity, consider the free space defined by the H-face of box B. It will not be broken into smaller spaces by the box A if box A and B align in H-axis. The larger the H-regularity, the fewer free spaces are broken into smaller pieces. L- and H-regularity are similarly defined to measure the ability to preserve the continuity of free spaces defined by the existing boxes' L- and H-faces.

In addition, larger H-regularity may lead to better support for future boxes. Consider the rectangle E in Fig. 11 and its adjacent area in the H-face of box B. They form a continuous support region for future boxes. The larger the continuous region, the more likely it can support future boxes. Therefore, H-regularity is given more weight when combined L-, W-regularity into a single measure,

$$\text{Regularity}(p) = \text{L-Regularity}(p) + \text{W-Regularity}(p) + \beta \times \text{H-Regularity}(p) \tag{10}$$

The ContactArea is defined by Zachariadis et al. (2012) to be the total contact area between the current box and its surrounding boxes. A larger contact area often results from denser local configurations. The SupportRatio is defined by Mack et al. (2004) as the total support area of the current box divided by its bottom area, which measures the stability of the packing layout.

Table 1
Dimensions of seven types of carton boxes employed by SF Express.

Type	L (cm)	W (cm)	H (cm)	Type	L (cm)	W (cm)	H (cm)
SF1	20	18	10	SF5	53	32	23
SF2	25	20	18	SF6	70	40	32
SF3	30	25	20	SF7	57	35	57
SF4	36	30	25				

5. Computational experiments

Our algorithms are implemented in Java (JDK 15.0.1, 64-bit) without explicit multi-threading. All experiments are conducted on laptops with an Intel i7-7920HQ CPU (3.1 GHz), 32 GB RAM, and 64-bit Windows 10 Home Edition. Our source code, test instances, and detailed computational results are available at https://github.com/ProfZHUWB/3DORP.git.

5.1. Generating DHRP instances

We generated a diverse set of problem instances reflecting typical scenarios to analyze the performance of our proposed robotic packing system and algorithms, considering box type, regularity of box assortments, distribution of box count among different types, and the number of boxes. The length, width, and height of loading spaces are set to 120 cm, 100 cm, and 150 cm, respectively.

Many courier service companies employ a collection of standard-sized carton boxes to pack customer orders. Table 1 shows the seven types of carton boxes used by SF Express Co., Ltd, one of the largest courier service companies in China.

From box type SF1 to SF4, two consecutive types share two common dimensions. For example, the smallest box type, SF1, shares two dimensions, 18 and 10, with the second smallest box type, SF2. Five SF1 boxes (after rotation) can be stacked into a space with dimensions $50 \times 20 \times 18$, the same as two SF2 boxes. Such regularity allows denser packing. We generated DHRP instances with the following regularity to study how regularity affects packing density,

- R0: two consecutive box types share no common dimension.
- R1: two consecutive box types share one common dimension.
- R2: two consecutive box types share two common dimensions.
- SF: the seven box types employed by SF Express, a mixture of R2, R1, and R0. SF5 and SF6 share one common dimension. SF4 and SF5 share no common dimension.

It is well-known that heterogeneity has a significant impact on volume utilization. Hence, logistic companies such as DHL and Amazon usually employ a small set of standard-sized carton boxes. Our choice of varying $TypeCount$ from 2 to 7 is based on the data from S.F., a large courier service provider in China. In addition, to cover application scenarios of various scales, the total box count ranges from 200, 500, and 1000, representing small, medium, and large scale applications, respectively.

The distribution of box count by different types is also considered to generate our DHRP instances. For example, companies that mostly sell consumer electronics will have many small boxes and a few large boxes. Therefore, Box types are divided into small, medium, and large according to their volume to simulate different scenarios. The smallest and largest $\lfloor \frac{TypeCount}{3} \rfloor$ box types by volume are considered small and large, respectively. The remaining box types are considered medium. Four distributions are considered,

- uniform: boxes are evenly drawn from each type.
- small, medium, large: 60% of boxes are evenly drawn from small, medium, and large types, respectively, and 40% of boxes are evenly drawn from other types.

Table 2
Key parameters in our proposed robotic packing system.

Category	Param	Description	Candidate values
algo	d	search depth in GLA	{0, 1, 2}
	e	search effort $e = k^d$, the number of states explored at depth d in GLA	{4, 9, 16, 25}
	β	weight of H-regularity in fitness function	{0.2, 0.5, 1.0, 2.0}
	$\|S\|$	number of simulated sequences in GLAS	{2, 4, 8}
sys	$\|\mathcal{O}\|$	number of open pallets	{1, 3, 5}
	N^b	number of boxes reachable by robotic arm	{1, 3, 5}
	N^k	number of boxes with known information	{5, 10, 20, 50}

An instance is named by concatenating the parameters that generate it, e.g., instance R2-3-1000-uniform is generated with type regularity R2, consisting of 3 box types and a total of 1000 boxes uniformly drawn from each box type. A total of $4\times6\times3\times4 = 288$ instances are generated, where dimensions of box types and arrival sequences are randomly generated.

5.2. Parameters and experimental strategy

Table 2 summarizes the key parameters. Category *algo* includes the parameters in our algorithm, which can be easily changed; category *sys* includes the parameters determined by hardware, which are hard to change once the system is deployed.

Under column "candidate values" are the promising values we identified based on our understanding of packing algorithms and many rounds of preliminary computational experiments. Our primary purpose is identifying parameters so that our packing algorithm works well for various application scenarios and hardware configurations. We would also like to analyze how system parameters impact the overall performance to provide some guidelines for practitioners who seek a robotic packing system.

The most comprehensive strategy is to try all possible combinations of all parameters and solve all 288 instances for each combination. There are $3\times4\times4\times3\times3\times3\times3\times4 = 5{,}184$ combinations of parameters, and we have to solve about 1.5 million instances, which is too costly. To reduce the overall cost of computational experiments, we employ a randomized strategy as follows. We create a total of 288 combinations of parameters, one for each instance. In every combination, the value of a parameter is uniformly randomly selected from its candidate values, and the value of each parameter is chosen independently. For example, since d has three possible values, each value will appear in approximately $288/3$ combinations. Similarly, parameter β has four values, and each value will appear in approximately $288/4$ combinations. The randomized strategy still permits the exploration of potential interaction between two parameters. For example, the interaction of $d = 1$ and $\beta = 0.2$ appears in approximately $288/12$ combinations.

5.3. Calibrating packing algorithm GLAS

We first determine the best parameter values in GLA. We found $d = 1$, $e = 16$, and $\beta = 1$ are the best choice through experiments. The detailed results and selection process is described in section 8.3 in the online supplements. We use these best values in all other experiments.

We then determine the best number of simulations $|S|$ in GLAS. For each of the 288 randomized combinations of parameters, GLAS is invoked three times with simulation count $|S|$ set to 2, 4, and 8, respectively. The results are summarized in Table 3, where column "avg util" is the average volume utilization, "#pallets" is the average number of pallets, and "tpb (s)" is the average time in seconds needed to load one box. ANOVA analysis shows that the impact of simulation count on average volume utilization is statistically significant with p value $2.96e-6$. The best choice is $|S| = 8$. T-test suggests that $|S| = 8$ outperform 2 and 4 with p value $4.19e-26$ and $6.04e-10$, respectively.

As the simulation count increases, the computational time increases proportionally, and the decision accuracy is also improved noticeably. If more powerful computers are available, or loading speed is not the primary factor, we can increase the simulation count further.

Table 3
Results for $|S|$ ($|\mathcal{O}| = 1$).

$\|S\|$	avg util	#pallets	tpb (s)
2	70.24%	18.60	0.54
4	72.62%	18.22	1.14
8	74.16%	17.98	2.39

Table 4
Results for $|\mathcal{O}|$ ($|S| = 0$).

$\|\mathcal{O}\|$	avg util	#pallets	tpb (s)	States
1	75.75%	17.85	0.33	181.9
3	70.80%	19.20	0.97	350.6
5	68.51%	19.81	1.31	407.0

5.4. Impact of system parameters

The next experiment investigates the impact of the number of open pallets on the performance of the semi-online GLA algorithm. For each of the 288 randomized combinations of parameters, GLA is invoked three times with open pallet count $|\mathcal{O}|$ set to 1, 3, and 5, respectively. The results are summarized in Table 4. ANOVA analysis shows that the impact of open pallet count on average volume utilization is statistically significant with p value 2.06e−21. The best choice is $|\mathcal{O}| = 1$. T-test suggests that $|\mathcal{O}| = 1$ outperforms 3 and 5 with p value 1.19e−35 and 7.59e−59, respectively.

Surprisingly, the average volume utilization drops substantially as the number of open pallets increases. In theory, more open pallets mean higher flexibility to place an incoming box, yielding better solutions. However, our GLA with limited $searchEffort$ may be misled to inferior placements by local fitness functions when there are too many candidates to choose from. To verify our hypothesis, we compute the total number of placements of all states explored divided by the total number of states when solving an instance. The average of over 288 instances are reported under the column "states" in Table 4.

During the tree search process, each child node represents the action of placing a box from the loading buffer at a feasible position on an open pallet. Certainly, increasing the number of open pallets augments the number of feasible positions for box placement, thereby increasing the number of child nodes. In the context of a complete tree search, a greater number of possibilities invariably leads to equivalent or superior solutions. However, to maintain reasonable computational time, our algorithm only explores the top k children of each node, as determined by our fitness measure. For incomplete tree searches, the additional child nodes introduced by extra open pallets could potentially "mislead" the search towards suboptimal solutions. Until a more effective fitness measure is developed that can leverage the advantage of extra child nodes, we recommend practitioners consider maintaining only a single open pallet.

The number of boxes reachable by a robotic arm N^b is mainly determined by the length of the robotic arm and the size of the carton boxes. Typically, it ranges from 1 to 5. Table 5 summarizes the experiment results for various N^b. ANOVA analysis shows that the impact of N^b on average volume utilization is statistically significant with p value 2.79e−16.

Table 5
Results for N^b ($|S| = 8$, $|\mathcal{O}| = 1$).

N^b	avg util	#pallets	tpb (s)
1	70.51%	19.17	2.11
2	73.63%	18.20	2.65
3	74.80%	17.78	2.86
4	75.63%	17.50	2.84
5	76.29%	17.31	2.95

Table 6
Results for N^k ($|S| = 8$, $|\mathcal{O}| = 1$).

N^k	avg util	#pallets	tpb (s)
5	71.60%	18.46	2.33
10	73.79%	18.01	2.41
20	74.69%	17.82	2.34
50	75.39%	17.76	2.29

As expected, the volume utilization improves as N^b increases since loading flexibility increases. The improvement from 1 to 2 is the most impactful and gradually diminishes as N^b increases. Large N^b is beneficial when hardware permits, though bigger robotic arms are usually more expensive. Increasing N^b from 2 to 5 does not change the computational time too much. Divide instances into three groups by the number of boxes, and we can observe a similar trend within each group, see Table 12 in the online supplements.

The number of boxes with known information, N^k, is mainly determined by the distance between the measuring gate and the robotic arm. Table 6 summarizes the experiment results for various N^k. ANOVA analysis shows that the impact of N^k on average volume utilization is statistically significant with p value 1.13e−06.

As expected, a larger N^k provides more information to our algorithm and increases the quality of the solution found. The improvement from 5 to 10 is the most impactful and gradually diminishes as N^k increases. In addition, this parameter does not affect the computational time of our algorithm much. However, larger N^k requires a longer conveyor belt and more land, increasing deployment costs. Within space limitations, we recommend using a larger N^k. Divide instances into three groups by the number of boxes, and we can observe a similar trend within each group, see Table 13 in the online supplements.

In summary, our dynamic setup with $N^k > 1$ and $N^b > 1$ is a meaningful extension to the classical online setup ($N^k = 1$ and $N^b = 1$). It significantly improves the quality of packing decisions by leveraging future knowledge and flexibility of reordering loading sequences.

5.5. Final results

Finally, we report the results of our calibrated algorithm on 288 instances with system parameters $|\mathcal{O}| = 1$, $N^b = 2$ and $N^k = 50$ as the benchmark for future research. The main reason to pick these values is for simplicity of deployment so that the robotic packing system can be deployed in many application scenarios. Many robotic arms available today are long enough to reach $N^b = 2$ boxes. It is relatively easy to ensure the conveyor belt between the measuring gate and the robotic arm is long enough to hold $N^k = 50$ boxes. We choose the following values for parameters in our GLAS algorithm, $d = 1$, $e = 16$, $\beta = 1$, $|S| = 8$ so that our algorithm can produce loading instructions for each incoming box within a few seconds on mainstream computers today. Companies with funding for high-performance computers can increase search effort e and simulation count $|S|$ to produce denser packing layouts.

Several trivial instances that require only a fraction of a pallet to load all boxes are removed from the 288 instances we generated. The detailed results for each non-trivial instance are reported in Table 16 in the online supplements.

Generally, bin packing algorithms tend to fully fill early pallets and leave the last pallet partially filled. In practice, the few items in the last "open" pallet can often be shipped the next batch. Therefore, the average volume utilization of closed pallets is more meaningful to practitioners and will be the main performance metric reported in this subsection. To verify that robotic arm constraints have a significant impact on volume utilization, we also implemented a manual loading version of our algorithm with robotic arm constraints disabled while respecting all other constraints and reported the results under the column "manual". If we only allow H-push and disable L- and W-push, the packing plan can be executed by simpler robotic arms, which are usually much cheaper. It is interesting to see how much performance will be sacrificed. We report the results of this version under the column "H-only".

We group results by regularity, box count, and box type count of an instance to see which characteristic affects our algorithm the most. The box regularity and the number of box types have the most significant impact, and the results are reported in Table 7 and Table 8, respectively. The box count has less impact and is reported in Table 15. In Table 7, column "avg util" is the average volume utilization of all pallets, including the last open pallet, while "avg util (closed)" excludes the last open pallet. As the regularity of box types increases from SD0 to SD2, the average volume utilization of closed pallets increases from 76.48% to 81.27%. It is interesting to note that the standard box sizes employed by SF Express produce a very dense layout with an average closed volume utilization of 80.52%. This is not surprising, as SF Express is one of the leading express service companies in China, with a solid technical team optimizing their operations. The last row gives the average results of all instances. There are also noticeable differences between the column "manual (closed)" and "avg util (closed)". The robotic arm constraints reduce the overall volume utilization by 3.07%. Comparing the column "H-only (closed)" with the "avg util (closed)" column, where L- and W-push are enabled, we can see that L- and W- push can enhance volume utilization by 2.76% overall, which is a meaningful improvement for practice. On average, the loading instruction per box can be computed under 3 s.

Table 8 shows the results grouped by the box type count. When the box type count increases from 2 to 7, the average volume utilization decreases from 84.76% to 75.41%. The difference between columns "manual (closed)" and "avg util (closed)", and the difference between column "H-only (closed)" and "avg util (closed)" exhibit similar patterns as in Table 7, confirming that both robotic arm constraints and L-, W-push have significant impact to volume utilization.

Fig. 12 illustrates the layout of the first closed pallet in four typical packing instances. The four instances all have 1000 boxes and two types of boxes; the main difference is box regularity. From (a) to (c), the chance of boxes neatly aligned with neighbors increases as box regularity increases, resulting in fewer "holes" and denser packing.

5.6. Effectiveness of heuristic rule on stability

We verified the stability of every box in the solutions to the 288 instances reported in the previous section. A total of 163,200 boxes are loaded in these instances. For every loaded box, we computed the convex hull of the contact regions in its bottom face as illustrated in Fig. 6 and checked if its center of geometry lies inside the convex hull. The center of geometry of all 163,200 boxes lies inside their convex hull. We also computed the shortest distance d_i between the center of geometry of box i to the boundaries of its convex hull. We divide $2d_i$ by the larger side of the bottom face of box i as a stability index. The stability index ranges from 0 to 1. When it is 0, the center of geometry lies on the boundary of the convex hull, and the box is very unstable; when the entire bottom face is supported, the stability index is 1, and the box is most stable. The average and min stability index of 163,200 boxes is 0.663 and 0.055, respectively.

We also implemented the SME model (1)–(4) as QCQP and solved it by IBM ILog Cplex 20 on a laptop equipped with Intel(R) Core(TM) i7-1065G7 CPU at a clock rate 1.30 GHz and 32 GB RAM. For every loaded

Table 7
Final results by box regularity.

Regularity	avg util	avg util (closed)	#pallets	tpb(s)	Manual (closed)	H-only (closed)
SD0	72.51%	76.48%	20.28	1.60	80.25%	74.96%
SD1	73.24%	77.88%	19.77	1.36	80.94%	74.88%
SD2	76.19%	81.27%	17.39	2.22	83.74%	76.94%
SF	75.93%	80.52%	16.35	1.99	83.47%	78.35%
Overall	74.48%	79.06%	18.43	1.80	82.13%	76.30%

Table 8
Final results by box type count.

Box type count	avg util	avg util (closed)	#pallets	tpb(s)	Manual (closed)	H-only (closed)
2	78.22%	84.76%	12.83	5.16	87.73%	81.06%
3	73.60%	80.01%	16.23	3.09	85.17%	76.45%
4	73.32%	79.40%	18.83	1.58	83.13%	75.83%
5	74.22%	77.30%	19.08	0.67	80.18%	75.40%
6	75.03%	78.20%	18.67	0.45	80.49%	75.90%
7	72.96%	75.41%	24.19	0.28	77.06%	73.77%
Overall	74.48%	79.06%	18.43	1.80	82.13%	76.30%

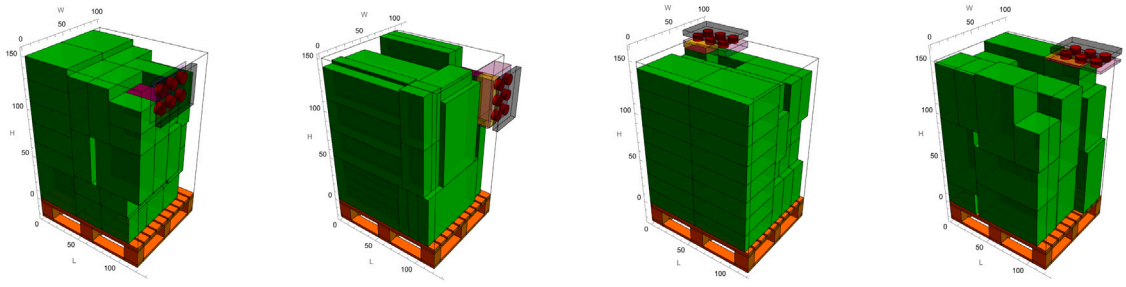

Fig. 12. First closed pallet layouts for various box regularities: (a) SD0; (b) SD1; (c) SD2; (d) SF.

Table 9
Statistics on 163,200 SME models.

	#var	#constraints	Time (ms)
min	12.0	14.0	0.0
max	14,052.0	11,534.0	216.0
avg	1046.0	880.7	9.8
std	1451.6	1190.8	13.7
Overall	74.68	17.97	2.58

box i, a SME model for all boxes up to box i in the same pallet is formulated and solved. Among 163,200 SMEs solved, 2,256 of them have no feasible solution. In other words, less than 1.4% of the layout where our heuristic rule predicted as stable is unstable. For most layouts, our simple heuristic rule agrees with the SME model. Table 9 reports statistics on 163,200 SME models solved, where columns '#var' and '#constraints' are the number of variables and constraints, respectively, and column 'time (ms)' is CPU time in milliseconds taken to solve a model, when the time is under milliseconds, it is reported as 0.0.

These computational results suggest that our simple heuristic rule is a pretty accurate approximation for stability. A layout indicated as stable by our rule is very likely to be stable. Our eye-ball inspection of the 3D visualization of packing layouts produced by our algorithm also looks stable.

6. Conclusion

Many courier companies have devoted much effort to automated material handling processes in their distribution center to reduce manpower costs and increase throughput. However, packing heterogeneous rectangular carton boxes into pallets is still done manually. We proposed a robotic packing system that attempts to automate this step. We model the two critical challenges in robotic packing systems, ensuring stepwise loading stability and avoiding collision between the robotic arm and placed boxes. The core of the robotic packing system is the online packing problem that decides how to pack incoming boxes with partial information. We devised a simulation-based greedy look-ahead algorithm to address this online packing problem.

We generated many diverse problem instances to emulate common application scenarios faced by many courier service companies. Comprehensive experiments show that our proposed algorithm can produce packing layouts with reasonable volume utilization within a practical time limit. The average volume utilization is only a few percent away from manual loading. The packing speed can reach a few seconds per box, matching that of skilled workers. Our system consists of mature hardware that is commercially available. We analyzed the impact of various parameters on the overall performance of a robotic packing system. We offered simple guidelines to help practitioners to configure their systems.

We have considerably simplified the operation of robotic arms to derive a simple collision-free model. The resulting model is reasonably accurate only for a particular type of robotic arm capable of loading relatively light carton boxes. There is a few possible research direction that can improve our approach. First, a split robot gripper is commonly utilized for lifting heavy boxes; it requires a very different model to avoid collisions. Second, when carton boxes are heavy, their weight distribution may significantly impact stepwise stable support constraints. How do we effectively incorporate weight distribution? Third, efficiently computing feasible placements given the packing layout of open pallets will considerably accelerate our approach.

Acknowledgments

This study is supported by the National Natural Science Foundation of China (Grant No. 71501075, U1901222), Guangdong Natural Science Funds for Distinguished Young Scholars (Grant No. 2015A030306007) and the Fundamental Research Funds for the Central Universities, China (Grant No. 2019ZD15).

Appendix A. Online supplements

The online supplements contain typical material handling processes in distribution centers and detailed computational results.

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ejor.2024.02.007.

References

Agarwal, M., Biswas, S., Sarkar, C., Paul, S., & Paul, H. S. (2021). Jampacker: An efficient and reliable robotic bin packing system for cuboid objects. *IEEE Robotics and Automation Letters*, *6*, 319–326.

Agha, H. N., Decamp, W. H., Shell, R. L., & Hall, E. L. (2000). Robotic palletizing of fixed-and variable-size/content parcels. In *Handbook of industrial automation* (pp. 673–687). CRC Press.

Ali, S., Ramos, A. G. a., Carravilla, M. A., & Oliveira, J. F. (2022). On-line three-dimensional packing problems: A review of off-line and on-line solution approaches. *Computers Industrial Engineering*, *168*.

Alonso, M. T., Alvarez-Valdes, R., & Parreño, F. (2020). A grasp algorithm for multi container loading problems with practical constraints. *4OR*, *18*, 49–72.

Bischoff, E. E., Janetz, F., & Ratcliff, M. (1995). Loading pallets with non-identical items. *European Journal of Operational Research*, *84*, 681–692.

Bortfeldt, A., & Wäscher, G. (2013). Constraints in container loading – a state-of-the-art review. *European Journal of Operational Research*, *229*, 1–20.

Chen, C., Lee, S.-M., & Shen, Q. (1995). An analytical model for the container loading problem. *European Journal of Operational Research*, *80*, 68–76.

de Azevedo Oliveira, L., de Lima, V. L., de Queiroz, T. A., & Miyazawa, F. K. (2021). The container loading problem with cargo stability: a study on support factors, mechanical equilibrium and grids. *Engineering Optimization*, *53*, 1192–1211.

De Castro Silva, J. L., Soma, N. Y., & Maculan, N. (2003). A greedy search for the three-dimensional bin packing problem: the packing static stability case. *International Transactions in Operational Research*, *10*, 141–153.

Demisse, G., Mihalyi, R., Okal, B., Poudel, D., Schauer, J., & Nüchter, A. (2012). Mixed palletizing and task completion for virtual warehouses. *Vol. 16*, In *Virtual manufacturing and automation competition (VMAC) workshop at the int. conference of robotics and automation*.

Elhedhli, S., Gzara, F., & Yildiz, B. (2019). Three-dimensional bin packing and mixed-case palletization. *INFORMS Journal on Optimization*, *1*, 323–352.

Fanslau, T., & Bortfeldt, A. (2010). A tree search algorithm for solving the container loading problem. *INFORMS Journal on Computing*, *22*, 222–235.

Galrão Ramos, A., Oliveira, J. F., Gonçalves, J. F., & Lopes, M. P. (2016). A container loading algorithm with static mechanical equilibrium stability constraints. *Transportation Research, Part B (Methodological)*, *91*, 565–581.

Ha, C. T., Nguyen, T. T., Bui, L. T., & Wang, R. (2017). An online packing heuristic for the three-dimensional container loading problem in dynamic environments and the physical internet. In *European conference on the applications of evolutionary computation* (pp. 140–155). Springer.

He, K., & Huang, W. (2011). An efficient placement heuristic for three-dimensional rectangular packing. *Computers & Operations Research*, *38*, 227–233.

Huang, W., & He, K. (2009). A caving degree approach for the single container loading problem. *European Journal of Operational Research*, *196*, 93–101.

Jiménez-Caballero, M., Conde, I., García, B., & Liarte, E. (2009). Design of different types of corrugated board packages using finite element tools. In *SIMULIA customer conference*.

Kocjan, W., & Holmström, K. (2008). Generating stable loading patterns for pallet loading problems. In *The fifth international conference on integration of AI and OR techniques in constraint programming for combinatorial optimization problems CPAIOR08*.

Mack, D., Bortfeldt, A., & Gehring, H. (2004). A parallel hybrid local search algorithm for the container loading problem. *International Transactions in Operational Research*, *11*, 511–533.

Martello, S., Pisinger, D., Vigo, D., Boef, E. D., & Korst, J. (2007). Algorithm 864: General and robot-packable variants of the three-dimensional bin packing problem. *ACM Transactions on Mathematical Software*, *33*, 7.

Nguyen-Vinh, K., Dewasurendra, H., Gonapaladeniya, S., Sarbahi, U., & Le, N. (2022). Stack algorithm implementation in robot-based mixed case palletizing system. In *2022 4th international conference on electrical, control and instrumentation engineering* (pp. 1–8). IEEE.

Parreño, F., Alvarez-Valdés, R., Oliveira, J. F., & Tamarit, J. M. (2010). Neighborhood structures for the container loading problem: a vns implementation. *Journal of Heuristics*, *16*, 1–22.

Parreño, F., Alvarez-Valdés, R., Tamarit, J. M., & Oliveira, J. F. (2008). A maximal-space algorithm for the container loading problem. *INFORMS Journal on Computing*, *20*, 412–422.

Saito, H., & Asari, Y. (2019). Semi-online three-dimensinal container loading problems. *IPSJ SIG Technical Report. 2019-AL-171*.

Schuster, M., Bormann, R., Steidl, D., Reynolds-Haertle, S., & Stilman, M. (2010). Stable stacking for the distributor's pallet packing problem. In *Intelligent robots and systems (IROS), 2010 IEEE/RSJ international conference on* (pp. 3646–3651). IEEE.

StatePostBureau (2020). State post bureau announces the operation of postal industry in 2019. http://www.spb.gov.cn/xw/dtxx_15079/202001/t20200114_2005598.html. Accessed August 5 2021.

Terno, J., Scheithauer, G., Sommerweiß, U., & Riehme, J. (2000). An efficient approach for the multi-pallet loading problem. *European Journal of Operational Research*, *123*, 372–381.

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

Wang, F., & Hauser, K. (2019). Stable bin packing of non-convex 3d objects with a robot manipulator. In *2019 international conference on robotics and automation* (pp. 8698–8704).

Wang, F., & Hauser, K. (2021). Robot packing with known items and nondeterministic arrival order. *IEEE Transactions on Automation Science and Engineering*, *18*, 1901–1915.

Wang, N., Lim, A., & Zhu, W. (2013). A multi-round partial beam search approach for the single container loading problem with shipment priority. *International Journal of Production Economics*, *145*, 531–540.

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, *183*, 1109–1130.

Wei, L., Oon, W.-C., Zhu, W., & Lim, A. (2011). A skyline heuristic for the 2d rectangular packing and strip packing problems. *European Journal of Operational Research*, *215*, 337–346.

Whiting, E., Ochsendorf, J., & Durand, F. (2009). Procedural modeling of structurally-sound masonry buildings. *ACM Transactions on Graphics*, *28*, 1–9.

Zachariadis, E. E., Tarantilis, C. D., & Kiranoudis, C. T. (2012). The pallet-packing vehicle routing problem. *Transportation Science*, *46*, 341–358.

Zhao, H., She, Q., Zhu, C., Yang, Y., & Xu, K. (2021). Online 3d bin packing with constrained deep reinforcement learning. *vol. 35*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 741–749).

Zhu, Q., Li, X., Zhang, Z., Luo, Z., Tong, X., Yuan, M., & Zeng, J. (2021). Learning to pack: A data-driven tree search algorithm for large-scale 3d bin packing problem. In *Proceedings of the 30th ACM international conference on information & knowledge management* (pp. 4393–4402).

Zhu, W., & Lim, A. (2012). A new iterative-doubling greedy–lookahead algorithm for the single container loading problem. *European Journal of Operational Research*, *222*, 408–417.

Zhu, W., Luo, Z., Lim, A., & Oon, W.-C. (2016). A fast implementation for the 2d/3d box placement problem. *Computational Optimization and Applications*, *63*, 585–612.

Zhu, W., Oon, W.-C., Lim, A., & Weng, Y. (2012). The six elements to block-building approaches for the single container loading problem. *Applied Intelligence*, *37*, 431–445.

Zudio, A., da Silva Costa, D. H., Masquio, B. P., Coelho, I. M., & Pinto, P. E. D. (2018). Brkga/vnd hybrid algorithm for the classic three-dimensional bin packing problem. *Electronic Notes in Discrete Mathematics*, *66*, 175–182, 5th International Conference on Variable Neighborhood Search.