Stable Bin Packing of Non-convex 3D Objects with a Robot Manipulator

Fan Wang[1] and Kris Hauser[1]

***Abstract*— Recent progress in the field of robotic manipulation has generated interest in fully automatic object packing in warehouses. This paper proposes a formulation of the packing problem that is tailored to the automated warehousing domain. Besides minimizing waste space inside a container, the problem requires stability of the object pile during packing and the feasibility of the robot motion executing the placement plans. To address this problem, a set of constraints are formulated, and a constructive packing pipeline is proposed to solve these constraints. The pipeline is able to pack geometrically complex, non-convex objects while satisfying stability and robot packability constraints. In particular, a new 3D positioning heuristic called Heightmap-Minimization heuristic is proposed, and heightmaps are used to speed up the search. Experimental evaluation of the method is conducted with a realistic physical simulator on a dataset of scanned real-world items, demonstrating stable and high-quality packing plans compared with other 3D packing methods.**

I. Introduction

Recent years have seen increasing interest in warehouse automation, including fully automatic robot packing, supported by the technical progress made in the field of robotic manipulation, as demonstrated by recent competitions like the Amazon Robotics Challenge. The current state of practice in fulfillment centers leaves the responsibility of container selection and packing to human worker intuition. Due to demanding schedules, workers cannot employ much foresight in the packing process and are reluctant to re-pack. This commonly results in grossly oversized containers that generate waste and high shipping costs (Fig. 1). Better containers and packing plans could be chosen using automated algorithms, whether packing is accomplished by humans or robots.

Problems that involve the placement of objects within a container or a set of containers are generally referred to as cutting and packing problems. Most existing packing algorithms apply to idealized scenarios, such as rectilinear objects and floating objects not subject to the force of gravity. To perform automatic packing in warehouses using a precomputed packing plan, several real-world issues need to be addressed, such as stability under force of gravity, and kinematics and clearance issues for the robot.

For a packing plan to be feasible with a robot manipulator, a comprehensive set of constraints need to be formulated. In addition to the two standard packing constraints:

1) *Noninterference.* Each object is collision free,
2) *Containment.* All objects are placed within the internal space of the container,

Fig. 1: Examples of poor space utilization in shipping boxes.

we introduce the following constraints necessary for a robot-packable plan:

3) *Stability.* Each object is stable against previously packed objects and the bin itself, and
4) *Manipulation feasibility.* A feasible robot motion exists to load the object into the target placement. The robot must obey kinematic constraints, grasp constraints, and collision constraints during this motion.

In the following sections, we refer to constraints 1 and 2 as the *non-overlap* constraints, and constraints 1-4 as all constraints, or the robot-packable constraints.

While the application of robot-packable constraints is independent of the particular packing problem addressed, this paper focuses on the problem of offline packing of 3D irregular shapes into a single container. To solve this problem under robot-packable constraints, we present the following main contributions:

1) A polynomial time constructive algorithm to implement a resolution-complete search amongst feasible object placements, under *robot-packable constraints*.
2) A 3D positioning heuristic named Heightmap-Minimization (HM) that minimizes the volume increase of the object pile from the loading direction.
3) A fast prioritized search scheme that first searches for robot-packable placement in a three-dimensional space that likely contains a solution, and falls back to search in a five-dimensional space.

Our algorithm and others in comparison are tested in a realistic physics simulator, by packing large quantities of itemsets using highly complex, real-world object scannings. With item sizes of 3-5 objects (e.g., a common Amazon order size), the success rate is 99.9% for finding and executing packing plans using small Amazon order boxes. Large number of items are also packed in stress tests, in these tests, 80% of the placement plans were successfully executed in the physics simulator, which is significantly better than the 17% success rate from a standard packing solver under the same testing condition. Empirical results also show that the new Heightmap-Minimization heuristic

*This work is partially supported by an Amazon Research Award.

F. Wang and K. Hauser are with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA {fan.wang2, kris.hauser}@duke.edu

finds more placements than existing heuristics.

II. Related Work

Most existing research on cutting and packing handles floating 2D and 3D rectilinear objects under the robot-packable constraints. Under some settings, such problems can be formulated and solved optimally using the exact algorithms. One example of these state-of-the-art exact algorithms is the solution to the 3D bin packing problem using branch and bound, proposed by Martello et al. [1], [2], whose work is further extended by many including Boef et al. [3] and Crainic et al. [4]. Heuristic methods and metaheuristic approaches have also been developed over the years, such as the popular Bottom-Left heuristic [5] and the Best-Fit-Decreasing heuristic [6].

More recent work has addressed irregular shape packing, and only heuristic methods are practical because the search space is infinite. Metaheuristics such as Simulated Annealing (SA) [7]–[9] and Guided Local Search (GLS) [10]–[14] are optimization methods that start with an initial placement and iteratively improve the placement by moving the pieces in the neighborhood while minimizing an objective function (e.g., overlap in the system). Other work has also proposed constructive positioning heuristics for 3D irregular objects, such as Deepest-Bottom-Left-Fill (DBLF), which places items in the deepest, bottom-most, left-most position; and Maximum Touching Area (MTA), which places an item in a position that maximizes the total contact area of its faces with the faces of other items [15].

Some research has taken aspects of stability into consideration during packing. Egeblad et al., for example, use a two-stage GLS packing algorithm that, in the first stage, optimizes for the center of gravity and inertia of the pile and, in the second stage, minimizes overlap in the system [11]; Liu et al. propose a constructive method that packs irregular 3D shapes using a Minimum-Total-Potential-Energy heuristic [9]. This method performs a grid search for the lowest gravitational center height Z for each placement. However, both proposed methods use heuristics only and do not verify the stability of each placement. In contrast from these works, our method enforces stability explicitly using constraints.

We also know of one packing work that takes into account robot manipulation feasibility [3], in which the author proposes a variant of the orthogonal 3D box packing scheme such that no prior packed box is in front of, to the right of, or above the current placing box, to avoid possible collision with a vacuum gripper. This placing rule, however, cannot guarantee collision-free placements with other gripper geometries and neglects robot kinematic constraints and graspability constraints.

To the best or our knowledge, ours is the first packing work to solve for stability and robot-feasibility constraints simultaneously. Moreover, these constraints can be solved for arbitrary shaped, complex 3D objects.

III. Problem Definition

We address the problem of offline packing of 3D irregular shapes into a single container while ensuring the stability of each packed item and feasibility of the placement with a robot gripper.

Specifically, for a set N geometries $\mathcal{G}_1, \ldots, \mathcal{G}_\mathcal{N}$ where $\mathcal{G}_i \subset \mathcal{R}^3$, let $\mathcal{C}$ donate the free space volume of the container and ∂C as the boundary of the free space. Let $T_i \cdot \mathcal{G}_i$ denote the space occupied by item i when the geometry is transformed by T_i. The problem is to find a placement sequence $S = (s_1, \ldots, s_N)$ of $\{1, \ldots, N\}$ and transforms $\mathcal{T} = (T_1, \ldots, T_N)$ such that each placement satisfies robot-packable constraint with geometries placed prior:

$$(T_i \cdot \mathcal{G}_i) \cap (T_j \cdot \mathcal{G}_j) = \emptyset,\ \forall i, j \in \{1, \ldots, N\}, i \neq j \quad (1)$$

$$T_i \cdot \mathcal{G}_i \subseteq \mathcal{C},\ \forall i \in \{1, \ldots, N\} \quad (2)$$

and for each $k = 1, \ldots, N$, stability constraints:

$$isStable\left(T_{s_k} \cdot \mathcal{G}_{s_k}, \mathcal{C}, T_{s_1} \cdot \mathcal{G}_{s_1}, \ldots, T_{s_{k-1}} \cdot \mathcal{G}_{s_{k-1}}\right) \quad (3)$$

and manipulation feasibility constraints:

$$isManipFeasible\left(T_{s_k} \cdot G_{s_k}, T_{s_1} \cdot \mathcal{G}_{s_1}, \ldots, T_{s_{k-1}} \cdot \mathcal{G}_{s_{k-1}}\right) \quad (4)$$

It is important to note that both stability and manipulation feasibility constraints must be satisfied for *every intermediate arrangement* of objects, not just the final arrangement.

A. Stability checking

Stability is defined as the condition in which all placed items are in static equilibrium under gravity and frictional contact forces. We model the stack using point contacts with a Coulomb friction model with a known coefficient of static friction. Let the set of contact points be denoted as $c_1, \ldots, c_K$, which have normals $n_1, \ldots, n_N$, and friction coefficients $\mu_1, \ldots, \mu_K$. For each contact c_k, let the two bodies in contact be denoted A_k and B_k. Let $f_1, \ldots, f_K$ denote the contact forces, with the convention that f_k is applied to B_k and the negative is applied to A_k. We also define m_i as the mass of object i, and cm_i as its COM. We take the convention that the container has infinite mass.

The object pile is in static equilibrium if there are a set of forces that satisfy the following conditions.

Force balance: $\forall i = 1, \ldots, N$,

$$-\sum_{k \mid i = A_k} f_k + \sum_{k \mid i = B_k} f_k + m_i g = 0. \quad (5)$$

Torque balance: $\forall i = 1, \ldots, N$,

$$-\sum_{k \mid i = A_k} (cm_i - c_k) \times f_k + \sum_{k \mid i = B_k} -(cm_i - c_k) \times f_k = 0.$$

Force validity: $\forall k = 1, \ldots, K$,

$$f_k \cdot n_k > 0, \quad (6)$$

$$\|f_k^\perp\| \leq \mu_k (f_k \cdot n_k). \quad (7)$$

where $f_k^\perp = f_k - n_k(f_k \cdot n_k)$ is the tangential component (i.e., frictional force) of f_k.

For a given arrangement of objects, an approximate set of contact points is obtained with the slightly scaled geometries in placement. A pyramidal approximation for the friction cone is used, and the conditions above are formulated as a linear programming problem over $f_1, \dots, f_N$, solved using the convex programming solver CVXPY [16]. If no such forces can be found, the arrangement is considered unstable.

B. Manipulation feasibility

This constraint checks feasibility of a packing pose when executed by a robot manipulator. This requires that the object be graspable from its initial pose and can be packed in the desired pose via a continuous motion, without colliding with environmental obstacles.

In our system, we limit ourselves to the existence of a feasible top-down placement trajectory within the grasp constraints, as robots performing pick and place (e.g., box packing) commonly use vertical motion [3]. We also assume the existence of a grasp generator that produces some number of candidate end effector(EE) transforms, specified relative to an object's geometry that may be used to grasp the object. The pseudo-code for this procedure is given in Alg. 1.

Algorithm 1: isManipFeasible

```
input : Desired placed geometry T·𝒢 and a set of
        grasp candidates {T_1^𝒢, ... T_n^𝒢}
1 for T^𝒢 ∈ {T_1^𝒢, ... T_n^𝒢} do
2 |   Compute top-down EE path 𝒫_ee interpolating from
  |     an elevated pose to a final pose T·T^𝒢;
3 |   for P_ee ∈ 𝒫_ee do
4 |   |   if ¬(IKSolvable (P_ee) ∧ inJointLimits(P_ee) ∧
  |   |     collisionFree(P_ee)) then Continue with Line1;
5 |   end
6 |   return True
7 end
8 return False
```

IV. Pipeline for Robot-packable Planning

We develop a constructive packing pipeline to solve for the set of robot-packable constraints proposed. Our algorithm accepts an itemset, a container dimension, a constructive positioning heuristic, and/or a packing sequence, to produce packing plans. The pipeline packs each item to its optimized feasible pose in sequential order, without backtracking.

Our pipeline primarily consists of 4 components, namely:

1) Placement sequence
2) Generate ranked transforms
3) Stability check
4) Manipulation feasibility check

The pipeline starts with a sequencing heuristic to sort all items in a tentative placement ordering and allocates them individually into the container in this sequence. For each object at the time of the allocation, a set of candidate transforms satisfying robot-packable constraints are generated and ranked based on the positioning heuristic. Constraint checks are performed in order until a transform satisfying all required constraints is returned.

A. Placement sequence

The placement sequence can be user-specified or generated by non-increasing bounding box volume rule. The generated sequence is subject to adjustment if a solution cannot be found in the specified ordering.

B. Generating ranked transforms

For a given item, a positioning heuristic (e.g., placement rule) identifies a free pose inside the container that is most preferred according to a specific criterion. Our pipeline accepts arbitrary positioning heuristics, but instead of applying the heuristic to obtain one optimal placement for each item, we use the score formulated from the positioning heuristic to rank candidate placements.

The candidate placements are obtained with a prioritized search among a discretized set of object poses. Instead of searching in the 6D space of SE(3), our algorithm first performs a grid search in a 3D space that likely contains robot-packable solutions. In the 3D search, the rolls and pitches of G are restricted to be a set of *planar-stable* orientations, which are a set of stable resting orientations of G on a planar surface, computed using the method of Goldberg et al. [17]. This speeds up the search for the common case of packing on the first layer and on horizontal supports. If no feasible solutions exist in the 3D space, the algorithm falls back to search in 5D, in which a grid search is performed for rolls and pitches as well.

The 3D search for collision-free placements of one object, given a set of rolls and pitches, is shown in Alg. 2. A grid search is performed for yaw, X, and Y at a given resolution, and the height Z of the placement is analytically determined as the lowest free placement. 2D heightmaps are used to accelerate the computation of Z to an efficient 2D matrix manipulation. Three heightmaps are computed: 1) a top-down heightmap H_c of the container and placed objects, 2) a top-down heightmap H_t of the object to be placed, and 3) a bottom-up heightmap H_b of the object to be placed. H_t and H_b are measured relative to the lower left corner of the orientated object. Raycasting is used to build these heightmaps, and rays that do not intersect with the object geometry are given height 0 in H_t and ∞ in H_b. The container heightmap is obtained once at the beginning of object placement search, and an object heightmap is computed once for each distinct searched orientation.

Given an object orientation and X, Y location, we calculate the lowest collision-free Z as follows:

$$Z = \max_{i=0}^{w-1} \max_{j=0}^{h-1} (H_c[x+i, y+j] - H_b[i,j]) \tag{8}$$

where (x, y) are the pixel coordinates of X, Y, and (w, h) to be the dimensions of H_t.

Once all collision-free candidate transforms are obtained, they are scored by a scoring function formulated from a positioning heuristic. For example, the Deepest-Bottom-Left-First heuristic can be formulated as the score:

$$Z + c \cdot (X + Y) \tag{9}$$

where c is a small constant.

Algorithm 2: 3DGridSearch

input : Geometry $\mathcal{G}$, container $\mathcal{C}$, rolls and pitches O
output: All collision-free candidate transforms

for $(\phi, \psi) \in O$ **do**
  for $\theta \in \{0, \Delta r, 2\Delta r, \ldots, 2\pi - \Delta r\}$ **do**
    Let $R \leftarrow R_z(\theta) R_y(\phi) R_x(\psi)$;
    Discretize legal horizontal translations of $R \cdot \mathcal{G}$ into grid $\{(X_1, Y_1), \ldots (X_n, Y_n)\}$;
    for (X, Y) *in* $\{(X_1, Y_1), \ldots (X_n, Y_n)\}$ **do**
      Find the lowest collision free placement Z at translation X, Y;
      $T \leftarrow (R, (X, Y, Z))$;
      if $T \cdot \mathcal{G}$ *lies within* $\mathcal{C}$ **then** Add T to $\mathcal{T}$;
    end
  end
end
return $\mathcal{T}$

Algorithm 3: packOneItem

input : item geometry $\mathcal{G}$, container $\mathcal{C}$, pitches and yaws O, sequence of the packed items $\{s_1, \ldots, s_i\}$, transforms of the packed items $\{P_1, \ldots, P_i\}$
output: Transform T or **None**

$\mathcal{T} \leftarrow$ 3DGridSearch($\mathcal{G}, \mathcal{C}, O$);
Score each T in $\mathcal{T}$ based on heuristic used;
for *up to* N *lowest values of* T *in* $\mathcal{T}$ **do**
  if $\neg$ *isStable(*$T \cdot \mathcal{G}, \mathcal{C}, P_1 \cdot \mathcal{G}_{s_1}, \ldots, P_i \cdot \mathcal{G}_{s_i}$*)* **then** **continue**;
  Compute grasp poses $T_1^{\mathcal{G}}, \ldots T_n^{\mathcal{G}}$ compatible with T;
  if *isManipFeasible(*$T \cdot \mathcal{G}$, $\{T_1^{\mathcal{G}}, \ldots T_n^{\mathcal{G}}\}$*)* **then** **return** T;
end
return ***None***

Algorithm 4: Robot-feasible packing with fall back procedures

input : Item geometries $\mathcal{G}_1, \ldots, \mathcal{G}_N$, container C, initial packing sequence $\{s0_1, \ldots, s0_N\}$
output: Transforms $\mathcal{T}$ and sequence S, or **None**

1 Initialize $\mathcal{T}, S, U, \mathcal{O}$ to empty lists;
2 **for** $\mathcal{G}_i \in \{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$ **do**
3   Get planar-stable rolls and pitches for $\mathcal{G}_i$ with the top n highest quasi-static probabilities $O_i = \{(\phi_1, \psi_1), \ldots, (\phi_n, \psi_n)\}$;
4   Add O_i to $\mathcal{O}$;
5 **end**
6 **for** *(*$s0_i \in \{s0_1, \ldots, s0_N\}$ **do**
7   $T =$ packOneItem(G_{s0_i}, C, O_{s0_i}, S, $\mathcal{T}$);
8   **if** T **then** Add T to $\mathcal{T}$, Add $s0_i$ to S;
9   **else** Add $s0_i$ to U;
10 **end**
11 **for** $u_i \in U$ **do**
12   Let $\{(\phi_1, \psi_1), \ldots, (\phi_n, \psi_n)\}$ be the planar-stable orientations in O_{u_i};
13   **for** $t_r \in \{0, \Delta r, 2\Delta r, \ldots, 2\pi - \Delta r\}$ **do**
14   **for** $t_p \in \{0, \Delta r, 2\Delta r, \ldots, 2\pi - \Delta r\}$ **do**
15       $O^t = \{(\phi_1 + t_r, \psi_1 + t_p), \ldots, (\phi_n + t_r, \psi_n + t_p)\}$;
16       $T =$ packOneItem(G_{u_i}, C, O^t, S, $\mathcal{T}$);
17   **if** T **then**
18   Add T to $\mathcal{T}$; Add u_i to S;
19   **continue** with Line 11
20   **end**
21   **end**
22   **return** ***None***
23 **end**
24 **return** $(\mathcal{T}, S)$

The candidates are then ranked by score (lower is better). If only robot-packable constraints are required, the placement candidate with the lowest score is returned. If additional constraints are specified, the ranked candidates will be checked for the additional constraints until a candidate satisfying all constraints is returned.

After a new object has been placed, we update the heightmap of the container H_c. This subroutine is also used in our heightmap minimization heuristic. Given a pose X, Y, Z of the object to be packed, and the top heightmap H_t at the given orientation, we calculate an updated heightmap H_c' adding the placed object as follows:

For all $i = 0, \ldots, w-1$, $j = 0, \ldots, h-1$, we let:

$$H_c'[x+i, y+j] = \max(H_t[i,j] + Z, H_c[x+i, y+j]) \quad (10)$$

if $H_t[i,j] \neq 0$, and otherwise

$$H_c'[x+i, y+j] = H_c[x+i, y+j]. \quad (11)$$

C. Pipeline summary and fall back procedures

A packing attempt for a single item is summarized in Alg. 3, and the overall pipeline for packing multiple objects is given in Alg. 4. Given a heuristic packing sequence, it calls Alg. 3 for each item with the set of planar-stable rolls and pitches. This first stage finds placements for most objects in typical cases. For the remaining unpacked items U, the algorithm activates the *fallback procedure*. The fallback procedure examines each unpacked item and attempts to perturb the planar stable orientations by iterating over rolls and pitches until a solution is found, and if no solution is found the algorithm terminates with failure.

V. Heightmap-Minimization Heuristic

The performance and solution quality of a multi-dimensional packing problem is highly susceptible to the item-positioning rule [18]. However, existing positioning heuristics for 3D packing are scarce and are commonly adapted directly from 2D packing, and therefore result in poor space utilization in the 3D container [4], [19]. To address these shortcomings, we propose a novel positioning heuristic called the Heightmap-Minimization (HM) heuristic, which favors item placements that result in the smallest

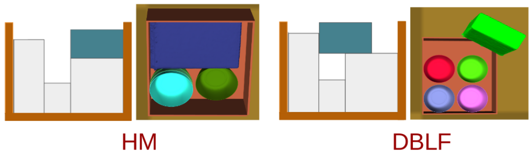

Fig. 2: Example packing placements obtained by HM and DBLF. HM finds more compact and stable packing compared to DBLF.

occupied volume in the container, as observed from the loading direction.

Specifically, HM scores a placement as follows. Given the candidate transform $T = (roll, pitch, yaw, X, Y, Z)$, compute a tentative container heightmap H'_c using the update routine described in Sec. 12. Suppose its shape is (w, h). The score for the placement using the HM heuristic is:

$$c \cdot (X + Y) + \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} H'_c[i, j] \tag{12}$$

where c is a small constant.

HM favors positions and orientations that result in good space utilization as it minimizes wasted space and holes that cannot be filled. HM also favors stable placements since the bottom of the object is encouraged to match the shape of the supporting terrain (Fig. 2).

VI. Experiment

We tested our algorithm on different item sets and validated plan feasibility in a physics simulator. Objects were drawn at random from a set of 94 real-world object meshes from the YCB [20] and the APC 2015 object set [21]. On average each mesh contains 10,243 vertices. Experiments are conducted on Amazon Web Services instance type m5.12xlarge. All computation times are measured on a single thread. Parameters used in the experiment are: heuristic constant $c = 1$; heightmap resolution 0.002m; step size in both X and Y 0.01m; $\Delta r = pi/4$ in range $[0, \pi)$; friction coefficient $\mu = 0.7$. Contact points are obtained using the exact geometry with a scale factor of 1.03. The top 4 planar-stable rolls and pitches with the highest quasi-static probabilities are used, and candidate number N = 100.

A. Robot manipulation feasibility with a vacuum gripper

The robot model used to verify robot feasibility constraints is a Staubli TX90 robot, equipped with a cylindrical vacuum gripper of 30 cm length and 2 cm diameter. The graspability constraints ask the vacuum gripper to grasp within a radius r = 2 cm in the horizontal plane to the object's center of mass when the object is sitting in flat orientations. The areas under the gripper should be solid planar areas (80% of the surface points directly below the tool are within 0.3mm to the estimated plane) for the vacuum opening to grasp normal to the surface, the resulting gripper axis needs to be within a tilting angle $\theta = \pi/4$ to the Z-axis.

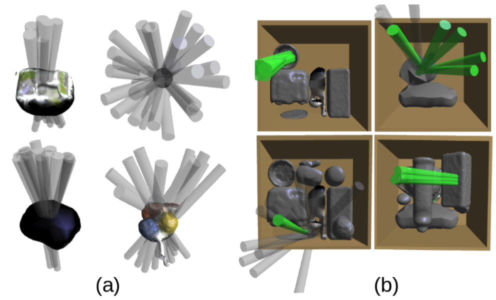

Fig. 3: (a) Grasp poses generated satisfying vacuum graspbility constraints. (b) Compatible gripper poses with candidate object orientation are checked for clearance with the container and the object pile. Collision-free grasps are shown in green and colliding grasps are colored in transparent grey.

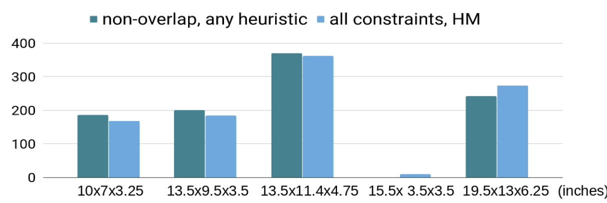

Fig. 4: Distribution of the solution containers found under different level of constraints. The x axis is the 5 container dimensions tested.

B. Small Order Packing

Simulating problem settings in a typical warehouse, we performed a small order packing test. Per communication with personnel at Amazon, 3-5 items are a standard order size. We generated 1000 random itemsets consisting of 3-5 models. The itemsets are verified with various testing methods to fit in at least one of the five containers used in the Amazon Robotics Challenge 2017 [22] under robot-packable constraints, and the smallest feasible container for each itemset is recorded.

We test our pipeline with HM heuristic and all constraints. Our algorithm seeks a feasible solution using the smallest container first, and if fails, the process is repeated on the second smallest container and so on until either a solution is found or all available containers are exhausted.

The success rate is 99.9% averaging 9.54 s per order. The only failure occurs when a large object needs to be tilted sideways to fit within a tight space, and no feasible vacuum grasp exists in the specified orientation. Fig. 4 compares the smallest bin statistics using all tested methods and ours. It appears that HM packing, despite adding all constraints, enlarges the container needed only marginally.

C. Comparisons on Large Itemsets

Next, we perform stress tests on itemsets of size 10. A tall container of size $32 \times 32 \times 30$ cm is chosen. 1000 itemsets of size 10 are generated and verified with all tested methods to have a non-overlap packing within the chosen container. Since the tilted gripper is likely to collide with the tall container chosen, we assume the gripper can grasp object of any orientation at the center of object's top surface, with the gripper axis vertically aligned to the Z axis.

Fig. 5: Examples of packing plans for itemsets of size 10.

TABLE I: Comparing planning techniques on 10-item orders with and without robot-packable constraints

	HM	DBLF [15]	MTA [15]	GLS [23]
Success, non-overlap (%)	99.9	98.4	88.9	78.9
Time, non-overlap (s)	15.7	14.2	14.1	502
Success, all constraints (%)	97.1	96.3	86.3	—
Time, all constraints (s)	34.9	50.1	95.4	—

We compare our HM heuristic against the DBLF and MTA heuristics [15], as well as an implementation of a guided local search (GLS) method as described by Egeblad et al. [11]. The fast intersection area theorem in Egeblad's paper was not implemented. Therefore, for the fairness of the comparison, GLS was run with 5 random restarts, and each restart was terminated after 300 s if a solution could not be obtained. GLS is also not tested for all constraints, as implementing robot-packable constraints in GLS methods is very challenging. Table I reports the percentage of solutions found and the average computation time.

Empirically, HM finds more solutions than any other method in comparison. With robot-packable constraint, HM finds 99.9% of all feasible solutions, leading the 2nd place DBLF heuristic by 1.5%, while MTA and GLS are not as competitive. After adding all constraints, each technique drops in success rate by a few percents, but HM still leads the other methods. The mean running time of HM is also 30% shorter, indicating that the highest ranked placements are more likely to be stable than the other heuristics.

In addition, only 3.2% (320 out of 10,000) of the items are packed with the fallback procedure, indicating the 3D space searched is indeed highly likely to contain robot-packable solutions. The fallback procedure is nonetheless important, as, with no fallback procedure, the success rate with all constraints drops from 97.1% to 72.4%.

With finer rotation granularity Δr and more candidates to check against robot-packable constraints, the success rate can be further improved at the cost of increased computation time (Table II).

TABLE II: Impact of Δr and candidate number N on the results

Rotation granularity Δr	$\pi/4$	$\pi/4$	$\pi/8$
Number of candidates N	100	500	500
Success, all constraints (%)	97.1	97.5	98.7
Time, all constraints (s)	34.9	70.05	89.40

TABLE III: Execution success rates in simulation, 10-item orders

	Success (%)	Drop (cm)	Horiz. Shift (cm)
Non-overlap constraint	17.11	1.95	1.29
All constraint	79.1	1.36	0.50

D. Executing Packing Plans in Simulation

Finally, we test the open-loop execution feasibility of packing plans in the Klamp't robot physics simulator [24]. In the simulation, the robot places one item after another using a top-down loading direction. The plan is considered a success if: 1) All items placed to the planned transforms without the robot and the objects colliding with items placed prior, and 2) all items contained within the container when placement is complete.

The robot used in the simulation is the TX90 robot model with the vacuum gripper described. We make the same assumption for the gripper as in the 10-item packing case. The robot places all items 1cm elevated from their planned transform; therefore there is an expected 1 cm drop. We allow 20 s for the items to settle before the next item is placed.

In the 3-5 items case, 100% of plans are executed successfully according to our success criteria. In the 10 item case, 768 out of 971 (≈80%) of robot-packable plans obtained with HM heuristic are executed successfully. This is significantly higher than the 17% success rate with robot-packable constraints. Further shown in Table III, items packed with all constraints undergo smaller displacements during packing execution, indicating increased stability.

The 20% failure cases are caused by an object falling out of its desired placement, which prevents subsequent items from being packed. The stability checker may be too optimistic, especially for intrinsically unstable objects like balls. Moreover, the impact of dropping an object could shift supporting objects.

VII. Conclusion

In this paper, we address the automated packing problem in a warehouse setting. A constructive pipeline is developed that can pack geometrically complex, non-convex objects with stability while satisfying robot constraints. A new Heightmap-Minimization heuristic is proposed as a positioning heuristic for efficient 3D irregular shape packing. Simulation results on exhaustive datasets demonstrate the effectiveness of the pipeline and the advantage of the new heuristic in finding stable and robot-packable plans. Robot-packable plans are shown to be far more successful in open-loop execution than non-overlap methods used in prior work.

Future work could address non-rigid objects or uncertainty in 3D scanned models. We may also be able to increase the execution success rate by implementing a more conservative stability check, or to perform a closed-loop execution.

REFERENCES

[1] S. Martello and D. Vigo, "Exact solution of the two-dimensional finite bin packing problem," *Management Science*, vol. 44, no. 3, pp. 388–399, 1998.

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

[3] E. den Boef, J. Korst, S. Martello, D. Pisinger, and D. Vigo, "Erratum to the three-dimensional bin packing problem: Robot-packable and orthogonal variants of packing problems," *Operations Research*, vol. 53, no. 4, pp. 735–736, 2005.

[4] T. G. Crainic, G. Perboli, and R. Tadei, "Extreme point-based heuristics for three-dimensional bin packing," *INFORMS Journal on Computing*, vol. 20, no. 3, pp. 368–384, 2008.

[5] B. Baker, E. Coffman, Jr., and R. Rivest, "Orthogonal packings in two dimensions," *SIAM Journal on Computing*, vol. 9, no. 4, pp. 846–855, 1980.

[6] D. Johnson, A. Demers, J. Ullman, M. Garey, and R. Graham, "Worst-case performance bounds for simple one-dimensional packing algorithms," *SIAM Journal on Computing*, vol. 3, no. 4, pp. 299–325, 1974.

[7] T. Kämpke, "Simulated annealing: Use of a new tool in bin packing," *Annals of Operations Research*, vol. 16, no. 1, pp. 327–332, Dec 1988.

[8] D. Zhang and W. Huang, "A simulated annealing algorithm for the circles packing problem," in *Computational Science - ICCS 2004*, M. Bubak, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 206–214.

[9] X. Liu, J.-m. Liu, A.-x. Cao, and Z.-l. Yao, "Hape3d—a new constructive algorithm for the 3d irregular packing problem," *Frontiers of Information Technology & Electronic Engineering*, vol. 16, no. 5, pp. 380–390, May 2015.

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

[11] J. Egeblad, "Placement of two and threedimensional irregular shapes for inertia moment and balance," vol. 16, pp. 789 – 807, 06 2009.

[12] C. Voudouris and E. P. K. Tsang, *Guided Local Search*. Boston, MA: Springer US, 2003, pp. 185–218.

[13] J. L. Viegas, S. M. Vieira, E. M. P. Henriques, and J. M. C. Sousa, "A tabu search algorithm for the 3d bin packing problem in the steel industry," in *CONTROLO'2014 – Proceedings of the 11th Portuguese Conference on Automatic Control*, A. P. Moreira, A. Matos, and G. Veiga, Eds. Cham: Springer International Publishing, 2015, pp. 355–364.

[14] J. A. Bennell, L. S. Lee, and C. N. Potts, "A genetic algorithm for two-dimensional bin packing with due dates," *International Journal of Production Economics*, vol. 145, no. 2, pp. 547 – 560, 2013.

[15] L. Wang, S. Guo, S. Chen, W. Zhu, and A. Lim, "Two natural heuristics for 3d packing with practical loading constraints," in *PRICAI 2010: Trends in Artificial Intelligence*, B.-T. Zhang and M. A. Orgun, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 256–267.

[16] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[17] K. Y. Goldberg, B. Mirtich, Y. Zhuang, J. Craig, B. Carlisle, and J. F. Canny, "Part pose statistics: estimators and experiments," *IEEE Trans. Robotics and Automation*, vol. 15, pp. 849–857, 1999.

[18] A. Lodi, S. Martello, and D. Vigo, "Tspack: A unified tabu search code for multi-dimensional bin packing problems," *Annals of Operations Research*, vol. 131, no. 1, pp. 203–213, Oct 2004. [Online]. Available: https://doi.org/10.1023/B:ANOR.0000039519.03572.08

[19] Lodi, Andrea, Martello, Silvano, and D. Vigo, "Heuristic algorithms for the three-dimensional bin packing problem," *European Journal of Operational Research*, vol. 141, pp. 410–420, 09 2002.

[20] Ycb benchmarks object and model set. [Online]. Available: http://ycbbenchmarks.org

[21] Rutgers apc rgb-d dataset. [Online]. Available: http://pracsyslab.org/rutgers_apc_rgbd_dataset

[22] Amazon robotics challenge official rules. [Online]. Available: https://www.amazonrobotics.com/site/binaries/content/assets/amazonrobotics/arc/2017-amazon-robotics-challenge-rules-v3.pdf

[23] J. Egeblad, B. K. Nielsen, and A. Odgaard, "Fast neighborhood search for two- and three-dimensional nesting problems," *European Journal of Operational Research*, vol. 183, no. 3, pp. 1249 – 1266, 2007.

[24] Klampt - intelligent motion laboratory at duke university. [Online]. Available: http://motion.pratt.duke.edu/klampt