IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 13, 2017, accepted April 10, 2017, date of publication May 10, 2017, date of current version June 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2702715

A Three-Dimensional Adaptive PSO-Based Packing Algorithm for an IoT-Based Automated e-Fulfillment Packaging System

TZUU-HSENG S. LI, (Member, IEEE), CHIH-YIN LIU, PING-HUAN KUO, NIEN-CHU FANG, CHENG-HUI LI, CHING-WEN CHENG, CHENG-YING HSIEH, LI-FAN WU, JIE-JHONG LIANG, AND CHIH-YEN CHEN
aiRobots Laboratory, Department of Electrical Engineering, National Cheng Kung University, Tainan 701, Taiwan

Corresponding author: Tzuu-Hseng S. Li (thsli@mail.ncku.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 105-2221-E-006 -104 -MY2 and in part by the Ministry of Education, Taiwan, within the Aim for the Top University Project through National Cheng Kung University, Tainan, Taiwan.

ABSTRACT With the development of online shopping and the demand for automated packaging systems, we propose an Internet of Things (IoT)-based automated e-fulfillment packaging system and a 3-D adaptive particle swarm optimization (PSO)-based packing algorithm. The proposed system leverages the IoT to connect the data collection and conversion layer, the packaging management layer, the decision-making layer, and the application layer. A cyber network connects each robot, sensor, and smart machine to achieve high velocity, flexibility of procedures, and real-time information exchange. When customers order merchandise online, the orders are received and rearranged, and the deployment of items in a box is planned by the system. The proposed packing algorithm controls the arrangement of items. It compares the size and volume of items and boxes to choose a box of suitable size, as well as deciding on the optimal arrangement of items. This algorithm solves the difficult 3-D Multiple Bin Size Bin Packing Problem (3-DMBSBPP) by integrating an adaptive PSO-based configuration algorithm. Our simulation results show that the packing algorithm can deploy items appropriately, with all items packed inside their box without overlap and with an overall center-of-gravity close to the bottom center of the box. When all the items cannot be packed into a single box, the proposed dividing strategies split the items into groups to pack into two or more boxes of similar size. Furthermore, comparing with the real packages we assessed, the proposed algorithm has a competitive performance. Lastly, our robotic experiments show that the proposed packing algorithm can be implemented and executed by a robot and a manipulator. It also demonstrates the efficiency of this system, in which all devices communicate well with each other and the robots accomplish the packaging task successfully and cooperatively.

INDEX TERMS 3-dimensional packing, adaptive-PSO, automated packaging system, industry 4.0, Internet of Things.

I. INTRODUCTION

Online shopping, or e-tailing, is defined as the sale of products to customers via the Internet [1] and is becoming more and more popular due to the advantages of its convenience, the variety of merchandise available, the ease with which price comparisons can be made, the lack of crowds, and the absence of boundaries of time or geography. The principles of successful e-commerce include short processing times, flexibility of systems and procedures, real-time information sharing, and so on. E-fulfillment distribution processes include slotting, picking, sorting, packaging, and delivering items [2]. Optimizing the distribution process is a key challenge for e-commerce as it includes labor-intensive and tedious tasks due to the high heterogeneity of items and the small

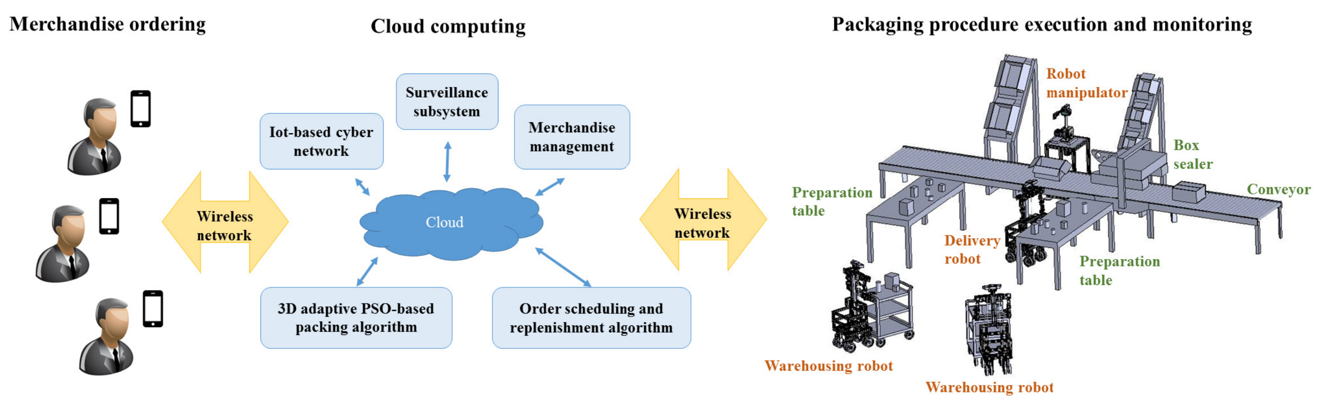


FIGURE 1. Application scenario for the proposed IoT-based automated e-fulfillment packaging system.

transaction sizes of individual customer orders [3]. To shorten fulfillment times and use less labor, many research and development efforts have been devoted to automation technologies.

Autonomous Guided Vehicles (AGVs) for the warehouse, such as Amazon Kiva robots [4] and Swisslog warehousing robots [5], can move shelving units to appointed places. To further the automatic selection of items and to pick up variously-shaped objects from shelves, robots need to be able to integrate visual perception and recognition, grasping motion planning, and online task planning [6], [7]. Amazon has even proposed an automated delivery system which uses miniature unmanned aerial vehicles (MUAVs) to deliver packages to customers [8]. All of this points towards an unstoppable tendency for the continued use of robots and other automatic technologies in e-fulfillment. Building an industrial sensor network which connects smart objects, mobile robots, and wireless sensors is a good way to improve information sharing and automation [9], [10].

In this paper, we propose an IoT-based automated e-fulfillment packaging system that leverages the IoT to connect customers, sensors, several kinds of robots, and intelligent algorithms. This reduces packaging times and costs while improving information exchange with customers. Our work is inspired by the concept of Industry 4.0, which was first proposed by the German government [11] and is associated with the basic concepts of the Internet of Things (IoT) [12], cyber-physical systems (CPS) [13], and smart factories [14]. All the components in the system will be connected and monitored by the IoT. The data collected from sensors, controllers, and enterprises will be converted into meaningful information and then analyzed and compared within the cyber network to extract useful insights and to generate a thorough picture of the monitored system [15]. As a result, the interconnection between physical assets and computational capabilities will be constructed and managed [16]. This new industrial revolution will make the whole manufacturing area become more transparent, automated, and intellectualized [17], which will lead to a movement from mass production to customized production. This will, in turn, make production more flexible and reduce unnecessary waste of resources [18].

The concept of the proposed IoT-based automated e-fulfillment packaging system is illustrated in Fig. 1. When customers order merchandise online, the orders will be received and computed in the cloud by intelligent algorithms. The computed results include defining the sequence in which orders should be fulfilled, and how items should be arranged in an appropriate box. This information will be communicated to each machine and robot in the distribution process through the IoT-based cyber network.

Several kinds of robots and automated machines are integrated into the system to cooperatively accomplish the packaging task. The warehousing robots choose items across multiple nearby orders and put them on preparation tables. The delivery robot passes items from a single order to the robot manipulator, which selects an appropriately sized box and puts the items into it. The box then passes to a box sealer along a conveyor. In this execution, each order is a specific task. The robots have to communicate with each other constantly to maintain a smooth process. Connecting all the machines and robots in a cyber network also provides opportunities to monitor the statuses of the robots and assign them suitable tasks, based on their statuses, to optimize the performance of the whole factory.

Also, in order to prevent the failure of a packaging task, a surveillance subsystem is incorporated, which monitors the status of the robots and the robot manipulator. When an unusual situation occurs, the system will sound an alarm and notify the engineers. The proposed system will improve the efficiency of e-fulfillment and reduce the labor that is necessary, while also providing customers with real-time information about their packages.

In the IoT-based automated e-fulfillment packaging system, in this paper, we propose a three-dimensional adaptive Particle Swarm Optimization (PSO)-based packing algorithm which mainly deal with the packing process of the distribution processes. The algorithm decides on the optimal arrangement of items as well as a packing sequence, and the results are sent to each robot in the proposed system.

The remainder of this paper is organized as follows. Section II presents the related works of packing problem. Section III introduces the framework and functions of the proposed IoT-based automated e-fulfillment packaging system. Section IV gives details of the 3D adaptive PSO-based packing algorithm and simulations using it are presented in Section V. In Section VI, implementation and experiments are presented. Finally, concluding remarks are given in Section VII.

II. RELATED WORK

In this section, the related work of the packing problem will be presented. The packing and cutting problem, in which a set of small geometric items are arranged into larger containing regions, has been a core area of research [19]. The objective of this problem is to find an efficient arrangement of items so as to best use resources, such as maximizing the use of containing regions or minimizing the packing cost. Wäscher et al. [20] developed a useful typology that partitions the problem according to objective, dimensionality, assortment of large containers, and assortment of small items. With different objectives, the packing problem can be divided into two parts: the bin packing problem and the loading problem. The objective of the bin packing problem is to pack all items into the minimum number of bins, while the objective of the loading problem is to fit the maximum number of items into a fixed set of containers [21].

Items can be divided by their shapes, both regular, such as rectangles and circles in 2D or cuboids and spheres in 3D, as well as irregular shapes. Packing irregularly-shaped items in 2D can be referred to as the nesting problem, and is usually used in garment manufacture, sheet metal cutting, furniture making, and shoe manufacture [22]. The circle and sphere packing problem can be treated as a max-min problem which tries to maximize the minimum distance between the centers of circular items [23], [24]. Some published researchers solve this problem by formulating it as a non-convex optimization problems [25], and some use optimization or heuristic approaches [26]–[28].

For rectangular items of variable sizes, the problem can be discussed across variable numbers of dimensions, from one-dimensional packing to three-dimensional packing. In the one-dimensional bin packing problem, the width of items is always identical to the bin, so only one dimension is relevant [29]. The best-known heuristics are the first fit decreasing (FFD) algorithm and the best fit decreasing (BFD) algorithm [30].

Two-dimensional packing problems are important in many industrial applications, such as cutting components from large sheets of material or arranging articles or advertisements in newspapers [31]. Due to the problems' practical importance in industry, many methods have been proposed to solve them and they can be found in the surveys by Hopper and Turton [32] and Lodi et al. [31].

The three-dimensional bin packing problem also arises frequently because of the need to store goods in warehouses and to transport them to customers [33]. Due to the potential heterogeneity of items and bins, Zhao et al. [21] partitioned the packing problem into six sub-problems. According to their categories, the packing problem in this paper is a Three-Dimensional Multiple Bin-Size Bin Packing Problem (3DMBSBPP), in which the items are strongly heterogeneous and the bins are weakly heterogeneous.

In most published research concerning the three-dimensional bin packing problem, bin size is kept consistent [34]–[36]. These researchers mainly focus on the development of placement heuristics, such as genetic algorithms [37], [38], tabu search algorithms [39], and other heuristic methods [40], [41].

In general, research on the 3DMBSBPP is scarce. Alvarez-Valdes et al. [42] proposed a GRASP/Path relinking algorithm for two- and three- dimensional multiple bin-size bin packing problems to choose the most appropriate bins so as to minimize transportation costs. First, the method sorts the items into bins, and then chooses the first item on the list and its corresponding bin. After fitting the first item into the bin with the left-bottom-corner principle, the residual space is calculated to choose the next item. When all the items are packed, the results are recorded as an initial solution; then the path relinking method exchanges items between bins and compares the solutions to improve the packing performance.

Both Brunetta and Grégoire [43] and Wu [44] deal with practical MBSBPPs in real factory scenarios. The factories have to pack small merchandise into boxes and then pack these boxes into larger bins. Both the boxes and the bins come in various sizes. Brunetta and Grégoire use a tree-search algorithm where each node of the search tree is solved using an extension of a pallet-loading heuristic [45]; Wu tests all possible bins with a container loading algorithm [46]. In other words, both of them deconstruct the multiple bin-size packing problem into several single-size bin problems.

Paquay et al. [47] also propose a mixed integer linear program to solve the MBSBPP; however, the main contribution of their research is in taking the weight distribution and center of gravity of each container into consideration to maintain safety during air transportation.

Unlike the existing 3DMBSBPP methods, we propose a 3D adaptive PSO-based packing algorithm that considers the positions and orientations of all items simultaneously and calculates the optimal arrangement of items. The proposed algorithm is also built for an e-fulfillment packaging system, so the objective is not only to optimize the filling rate: customer convenience and safety are also taken into consideration. The contributions of this paper are as follows. Firstly, we propose a new 3D adaptive PSO-based packing algorithm for solving the difficult 3D multiple bin size bin packing problem. The algorithm can deal with strong heterogeneity of items and variable box sizes.

Secondly, the packing algorithm can be implemented in an automated e-fulfillment packaging system. The algorithm considers the real situations of the robots and the robot manipulator to ensure a smooth process. Furthermore, the customer convenience and safety are also taken into consideration, such that the maximal size and weight of boxes are limited and, for multiple-box deliveries, similar box sizes are preferred.

Finally, we propose our own IoT-based automated e-fulfillment packaging system to integrate the distribution process with the algorithm and achieve the components of successful e-commerce, including short processing times, flexibility of systems and procedures, and real-time information sharing.

III. IoT-BASED AUTOMATED e-FULFILLMENT PACKAGING SYSTEM

Packaging is an important activity in distribution systems and supply chains [48]. Good packaging ensures the safe and efficient delivery of an item, in sound condition, to the ultimate consumer [49]. A good packaging system in an e-fulfillment warehouse has to be fast, flexible, and low-cost, and should include good customer communication [2]. To achieve these aims, we propose an integrated and automated packaging system that leverages an IoT-based cyber network structure to connect sensors, machines, and robots.

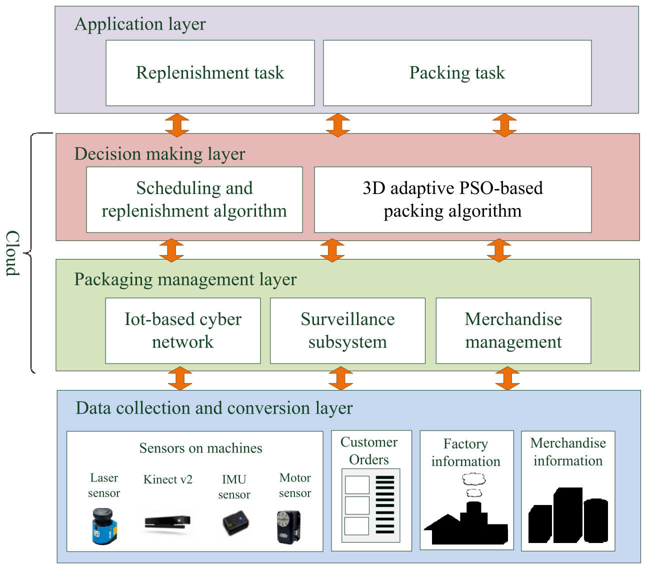


FIGURE 2. Four-layer structure of the proposed integrated autonomous packaging system.

A. FRAMEWORK OF THE SYSTEM

The network architecture of the proposed IoT-based automated e-fulfillment packaging system is shown in Fig. 2. It consists of four layers: the data collection and conversion layer, the packaging management layer, the decision making layer, and the application layer.

The data collection and conversion layer connects the sensors to the machines, the customer orders, the factory information, and the merchandise information. The sensors allow the robots to recognize merchandise, to move in the factory environment, and to detect unusual situations. A factory environment map and the merchandise information can be collated autonomously by robots or be constructed mutually with human beings. In the data collection layer, customer orders are received, stored, and managed for packaging. The collected data is first computed locally and then converted into useful information which is transmitted to the next layer.

In the packaging management layer, the IoT-based cyber network connects each robot and machine. It is built and updated according to the working status feedback of the robots. Furthermore, merchandise information, such as quantities, images, and sizes can be automatically synchronized between all devices in the network. This leads to two main advantages; first, the network makes the updating of merchandise information easier. A user can add or delete an item on any device in the system and the information is shared throughout the cyber network. All related algorithms are updated to adapt to the changed situation. The robots and robot manipulator receive this updated information, which ensures that they still accomplish their tasks accurately. Secondly, the network helps with stock management. The quantities of every kind of merchandise are monitored to manage stock. And the consumption data can further indicates stock turnover as well as level.

The intelligent algorithms in the decision making layer decide how the devices in the system perform. The scheduling and replenishment algorithm calculates the schedule of orders and the replenishment sequence. The packing algorithm decides the size of box that should be chosen, and how best to arrange the merchandise in the box. These calculated results are transmitted to the robots and the robot manipulator, which execute the packaging task in the application layer.

A surveillance subsystem is another important part of this system. It monitors the robots and the environment, through information transmitted from sensors, to detect abnormal situations anywhere in the process. If the subsystem detects excessive vibration or an unusual movement of motors on a robot, both of which may result in task execution failure, the surveillance subsystem will issue a warning and shut down the corresponding robot immediately to allow for human correction.

B. FLOWCHART OF THE SYSTEM

The flowchart of the proposed system is shown in Fig.3. First, orders are received and arranged by the scheduling and replenishment algorithm, which arranges the orders by comparing their contents. Orders containing similar items are collected together to reduce the picking time and the moving path of the warehousing robots.

After scheduling, the process is divided into three parallel threads: a packing thread, a replenishment thread, and a monitoring thread. To pack all items into a box appropriately, the proposed 3D adaptive PSO-based packing algorithm calculates the total volume of the items and chooses a suitably sized box while arranging the deployment of items in the box with the packing sequence. The calculated results are then

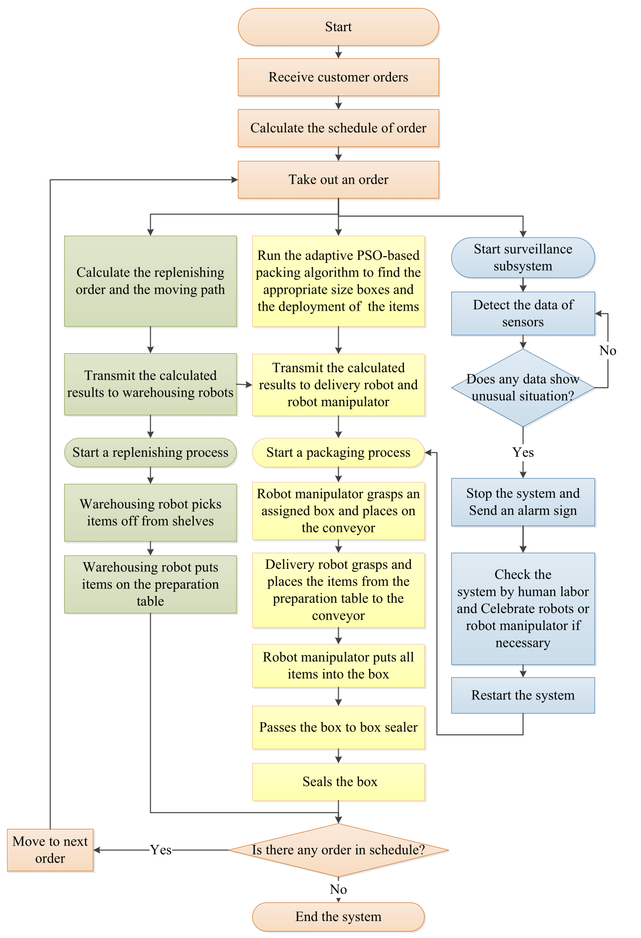

Start
Receive customer orders
Calculate the schedule of order
Take out an order
Calculate the replenishing order and the moving path
Run the adaptive PSO-based packing algorithm to find the appropriate size boxes and the deployment of the items
Start surveillance subsystem
Detect the data of sensors
Transmit the calculated results to warehousing robots
Transmit the calculated results to delivery robot and robot manipulator
Does any data show unusual situation?
No
Yes
Start a replenishing process
Start a packaging process
Warehousing robot picks items off from shelves
Robot manipulator grasps an assigned box and places on the conveyor
Stop the system and Send an alarm sign
Warehousing robot puts items on the preparation table
Delivery robot grasps and places the items from the preparation table to the conveyor
Check the system by human labor and Celebrate robots or robot manipulator if necessary
Robot manipulator puts all items into the box
Restart the system
Passes the box to box sealer
Seals the box
Move to next order
Yes
Is there any order in schedule?
No
End the system

FIGURE 3. **Flowchart of the proposed integrated autonomous packaging system.**

sent to the delivery robot and robot manipulator, respectively. Next, the robot manipulator places the assigned box on the conveyor while the delivery robot picks up all items from the preparation table and moves them to the conveyor. Then the robot manipulator packs all the items into the box. When all the items are packed, the box will be passed to the box sealer.

C. INTELLIGENT ALGORITHMS OF THE SYSTEM

1) SCHEDULING AND REPLENISHMENT ALGORITHM

In a make-to-order business model, products are custom-made and are delivered to customers directly from the factory within a very short time. As a result, production must be well scheduled to optimize delivery speeds. [50]

For the proposed e-fulfillment packaging system, the schedule of orders will influence the efficiency of the replenishment and packaging process. The warehousing robots are responsible for picking items off of shelves and putting them on the preparation table. The items on the preparation table may belong to several orders, so the delivery robot has to recognize the items of the order currently being executed and pass them to the robot manipulator. Thus, the efficiency of the warehousing robots will influence the efficiency of the packaging process. Therefore, the scheduling algorithm has to take both the contents of each order and the position of each item into consideration.

2) 3D ADAPTIVE PSO-BASED PACKING ALGORITHM

Packing all items into appropriately sized boxes is a three-dimensional multiple bin-size bin packing problem. In addition to considering the arrangement of items in a box, a good packing algorithm has to be viewed as part of the packaging system and reflect the functions of packaging.

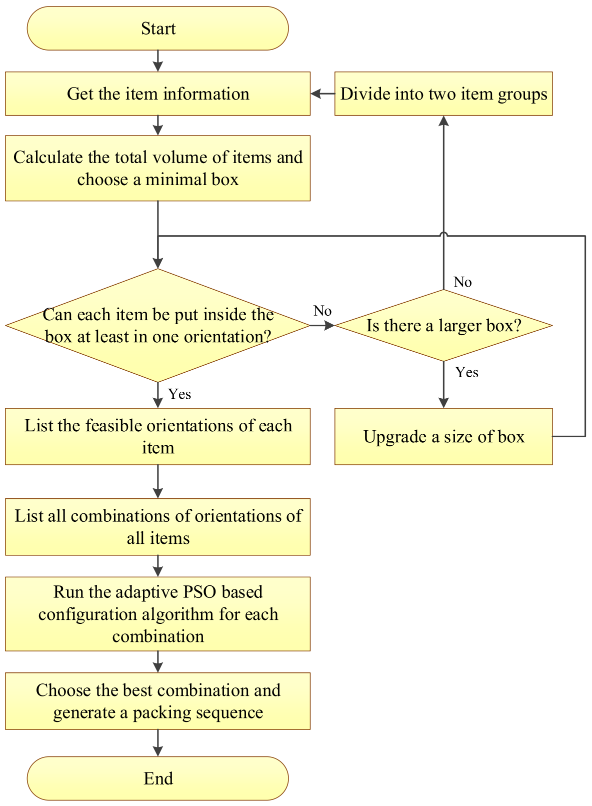
The functions of packaging can be categorized as follows: protection, promotion, communication, convenience, apportionment, and volume and weight efficiency [51]. Convenience denotes the ability of an item to be handled, which, together with the volume and weight design, will affect the efficiency of transportation and the convenience and safety of the customer. Packages that are too heavy or large are hard to transport, making it more likely that merchandise will be damaged [52]. Therefore, the size and weight of boxes have to be appropriate for handling by a human being. Furthermore, when items belonging to the same order cannot be packed into a single box and have to be divided into two or more boxes, a large diversity in box size will make transportation more difficult. Using boxes of similar sizes within each order is better. Based on these considerations, we propose a 3D adaptive PSO-based packing algorithm, which will be described in next section.

IV. 3D ADAPTIVE PSO-BASED PACKING ALGORITHM

The packing task consists of two parts: packing all items into a box, and dividing items into two or more boxes when everything cannot be packed into just one box. We utilize an adaptive PSO-based configuration algorithm for packing items, and three strategies for dividing items. The flowchart is shown in Fig. 4. The packing algorithm chooses a suitable box by calculating the total volume of the items, and then checks that all items can be placed into the box. Next, an adaptive Particle Swarm Optimization (PSO) [53] algorithm is used to decide the position of each item in the box. In the adaptive PSO algorithm, both adaptive inertia weight and adaptive acceleration coefficients can be tuned. The main advantage of the adaptive PSO is its high convergence speed. Comparing with other algorithms, it can converge in much fewer iteration with high accuracy. Convergence speed is an important factor in real-time applications, especially our packing algorithm has to calculate many kinds of combinations to find out a better one.

The packing algorithm includes five steps: 1) Choosing a minimally-sized box, 2) Checking that all items can be put into the box and revising box choice if necessary, 3) Listing all feasible orientations of each item and all combinations of orientations of all items, 4) Running the configuration algorithm to obtain the positions of items from the first combinations, 5) Choosing a combination and generating a packing sequence.

The minimally-sized box is chosen according to the total volume of the items. To simplify the computation, each piece



Start
Get the item information
Divide into two item groups
Calculate the total volume of items and choose a minimal box
Can each item be put inside the box at least in one orientation?
No
Is there a larger box?
No
Yes
Yes
List the feasible orientations of each item
Upgrade a size of box
List all combinations of orientations of all items
Run the adaptive PSO based configuration algorithm for each combination
Choose the best combination and generate a packing sequence
End

FIGURE 4. Packing Flowchart.

of merchandise is approximated as a cuboid with a center, $X_i = (x_i^m, y_i^m, z_i^m)$, and with a length, width, and height (l_i^m, w_i^m, h_i^m).We find a minimal box, $b_{\min}$, by calculating the total volume of the items, along with proper buffer space, as follows:

$$b_{\min} = \arg\min_{b_j}(w \sum\nolimits_{i=1}^{I} V_i^m < V_j^b) \tag{1}$$

where V_i^m is the volume of the piece of merchandise m_i, $i = (1, \ldots, I)$, V_j^b is the volume of the box b_j, $j = (1, \ldots, J)$, and w is a regulation weight used to add an additional proper buffer space.

Next, we test whether all items can be put inside the box. For each item, there are six placing orientations that are parallel to the three edges of the box. Some examples are shown in Fig. 5. For each placing orientation, if the three edges of the item are all smaller than the three edges of the box, this orientation will be recorded as feasible. All six orientations are tested with the formula:

$$o_k^m = \begin{cases} 1, & l_i'^m < l_j^b \cap w_i'^m < w_j^b \cap h_i'^m < h_j^b \\ 0, & otherwise \end{cases}, k = 1, \ldots 6 \tag{2}$$

where $l_i'^m, w_i'^m, h_i'^m$ are the rotated edges of the item. Once the item changes its orientation, the system recalculates the edges for future use.

After testing each orientation, the system calculates the number of feasible orientations using (3) and determines whether the item can be placed into the box or not by creating a binary variable, O_i, in (4). If the value of variable O_i is 1, the item can be put inside the box in at least one dimension.

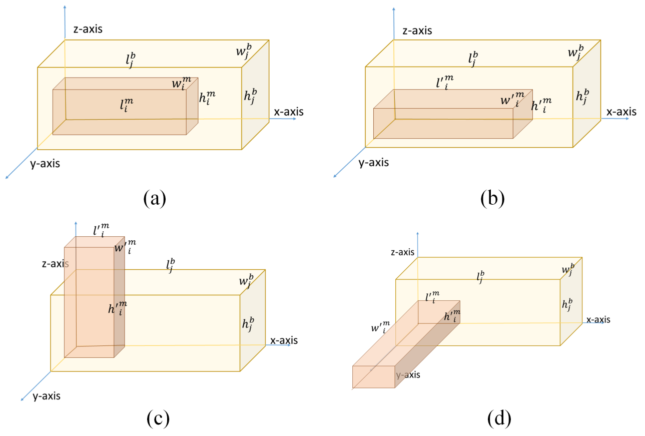


FIGURE 5. Examples of item placement orientations.

$$o_i^m = \sum_{k=1}^{6} o_k^m \tag{3}$$

$$O_i = \begin{cases} 1, & otherwise \\ 0, & o_i^m = 0 \end{cases} \tag{4}$$

We can sum the variable O_i for all items; if the summed value is equal to the number of items, as $\sum_{i=1}^{I} O_i = I$, we are assured that all items can be put inside the box. If any item cannot be put into the box, the system will upgrade the size of box and recalculate (2)-(4) until all items can be put into the box. An appropriate box size can be found as follows:

$$b_{app} = \arg\min_{b_j}(\sum\nolimits_{i=1}^{I} O_i = I) \tag{5}$$

If there is no larger box, the order will be separated into two groups and then recalculated. After ensuring that all items can be put inside the box(es), the system lists all feasible orientations of all the items and generates an orientation combination list, containing q_c combinations, where q_c can be calculated by:

$$q_c = o_1^m \cdot o_2^m \ldots\ldots o_I^m \tag{6}$$

For example, if an order contains three items that can be put in a box in 4, 3, and 2 orientations, respectively, there are 24 combinations on the list. For each combination, the system runs an adaptive PSO-based configuration algorithm to find the optimal arrangement of items.

A. ADAPTIVE PSO-BASED CONFIGURATION ALGORITHM

In this algorithm, each particle is encoded with the center positions, edges, velocities, and weights of each item, along with a fitness value and its best local solution, as (7). In addition, another particle "gbest" is created to store the global best solution during computing.

$$p_n = (X_i, V_i, l_i^m, w_i^m, h_i^m, w_i, f, pbest_n, f_{pbest}), i = 1, \ldots.I \tag{7}$$

where X_i and V_i are the center position and velocity of the ith item, and w_i is the weight of the ith item. The item center positions are updated by:

$$V(t+1) = w_a(t)\cdot V(t) + a_1 \cdot rand_1 \cdot (pbest_n - X(t)) + a_2 \cdot rand_1 \cdot (gbest_n - X(t)) \tag{8}$$

$$X(t+1) = X(t) + V_i(t+1) \tag{9}$$

where $X(t)$ and $V(t)$ represent the position and velocity, respectively, of all items at time t, $pbest_n$ is the best local solution of the nth particle, and $gbest_n$ is the global best solution of all the particles. $w_a(t)$ is the inertial weight, a_1 is the weight of the distance between the present position and the local best solution, and a_2 is the weight of the distance between the present position and the global best solution.

The inertial weight,$w_a(t)$, determines the size of the search area and the convergence rate. If the value of $w_a(t)$ decreases, the search area will become smaller and it will be easier for the algorithm to converge. a_1 and a_2 adjust according to the exploration state and the exploitation state. With a higher a_1, the search area will expand. On the other hand, the search area will centralize to the global best solution when a_2 is high.

These three weights are adjusted in real time according to the search results. The algorithm examines whether a particle is updated or not by the following equation.

$$S(n,t) = \begin{cases} 1, & \text{if } fit(pbest_n^t) > fit(gbest_n^t) \\ 0, & \text{if } fit(pbest_n^t) \le fit(gbest_n^t) \end{cases} \tag{10}$$

If particle n has updated, the value of function S is 1. All the examined results can be summed up with an update proportion using the formula:

$$P_s(t) = \frac{\sum_{n=1}^{N} S(n,t)}{N} \tag{11}$$

After this, $w_a(t)$, a_1, and a_2 are defined according to the following equations.

$$w_a(t) = (w_{\max} - w_{\min})P_s(t) + w_{\min} \tag{12}$$

$$\begin{cases} a_1 = \alpha \cdot (P_s(t)) + \beta \\ a_2 = \alpha \cdot (1 - P_s(t)) + \beta \end{cases} \tag{13}$$

During the early iterations, larger values for $w_a(t)$ and a_1 compel the particles to search in all areas; thus, the chances of finding a good solution increase. When the update proportion decreasing, the value of $P_s(t)$ increases, followed by the value of a_2. A high a_2 and a low $w_a(t)$ compel the particles to centralize on a global solution and speed up the convergence. The adjustment of the three weights can lead to better searching performance. The value of $w_{\max}$ and $w_{\min}$ are set as 1 and 0, respectively, so the range of the adaptive inertia weight is [0, 1]. The adaptive acceleration coefficients are adjusted by α and β, where α is set as 2.05 and β is set as 0.5.

After updating the center positions of the items, the system then checks whether any item is beyond the scope of the box by comparing the vertex coordinates of the box and each item. The vertex coordinates of a box and an item are shown in Fig. 6. The center position of a box, b_j, is defined as X_j, where $X_j = (x_j^b, y_j^b, z_j^b)$. The length, width, and height of the box are l_j^b, w_j^b, and h_j^b. The bottom left vertex of the box is the origin point for the coordinate geometry.

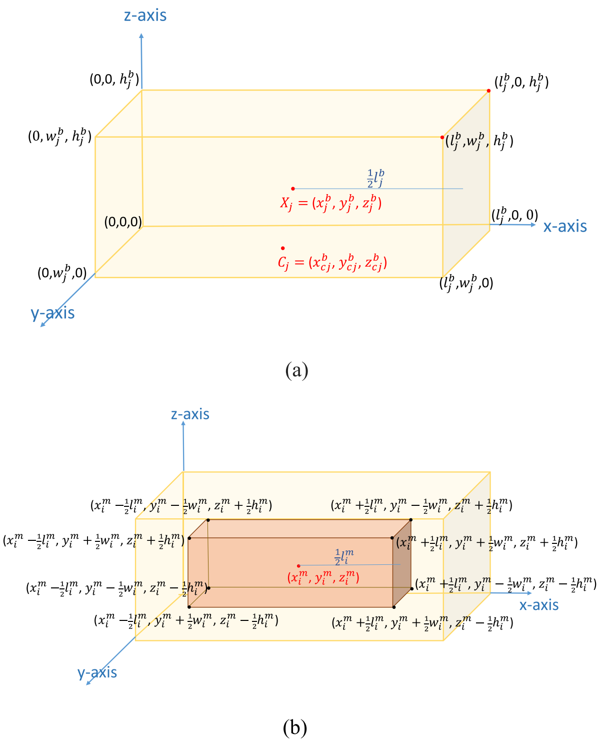

z-axis
$(0,0,h_j^b)$
$(l_j^b,0,h_j^b)$
$(0,w_j^b,h_j^b)$
(l_j^b,w_j^b,h_j^b)
$\frac{1}{2}l_j^b$
$X_j = (x_j^b, y_j^b, z_j^b)$
$(0,0,0)$
$(l_j^b,0,0)$
x-axis
$C_j = (x_{cj}^b, y_{cj}^b, z_{cj}^b)$
$(0,w_j^b,0)$
$(l_j^b,w_j^b,0)$
y-axis
(a)
z-axis
$(x_i^m - \frac{1}{2}l_i^m, y_i^m - \frac{1}{2}w_i^m, z_i^m + \frac{1}{2}h_i^m)$
$(x_i^m + \frac{1}{2}l_i^m, y_i^m - \frac{1}{2}w_i^m, z_i^m + \frac{1}{2}h_i^m)$
$(x_i^m - \frac{1}{2}l_i^m, y_i^m + \frac{1}{2}w_i^m, z_i^m + \frac{1}{2}h_i^m)$
$(x_i^m + \frac{1}{2}l_i^m, y_i^m + \frac{1}{2}w_i^m, z_i^m + \frac{1}{2}h_i^m)$
$\frac{1}{2}l_i^m$
(x_i^m, y_i^m, z_i^m)
$(x_i^m - \frac{1}{2}l_i^m, y_i^m - \frac{1}{2}w_i^m, z_i^m - \frac{1}{2}h_i^m)$
$(x_i^m + \frac{1}{2}l_i^m, y_i^m - \frac{1}{2}w_i^m, z_i^m - \frac{1}{2}h_i^m)$
x-axis
$(x_i^m - \frac{1}{2}l_i^m, y_i^m + \frac{1}{2}w_i^m, z_i^m - \frac{1}{2}h_i^m)$
$(x_i^m + \frac{1}{2}l_i^m, y_i^m + \frac{1}{2}w_i^m, z_i^m - \frac{1}{2}h_i^m)$
y-axis
(b)

FIGURE 6. Vertex coordinates of a box and an item. (a). Vertex coordinates of a box (b). Vertex coordinates of an item.

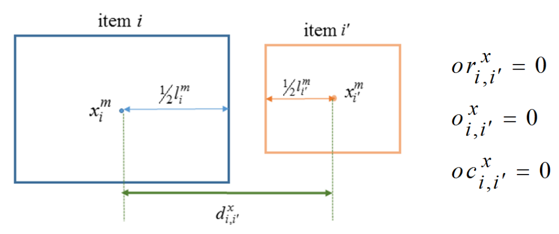
The system compares the boundaries separately in three dimensions. Take the x-dimension as an example; the box boundaries are 0 and l_j^b, and the item boundaries are $x_i^m \pm 1/2l_i^m$. When all the item boundaries are between the box boundaries, we can be sure that this item is inside the box. On the other hand, if a part of the item is outside of the box the system can then adjust the item's center using:

$$x_i^m = x_j^b - 1/2l_j^b + 1/2l_i^m \tag{14}$$

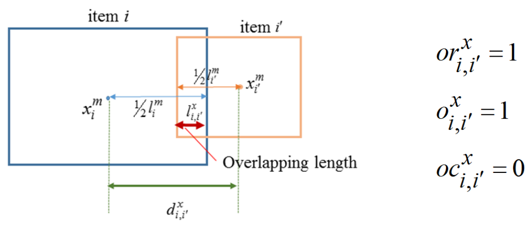
Therefore, after adjustment, the item will be just inside the box boundaries. The system checks every item in each dimension in the same way to ensure that all items are inside the scope of the box. When all item positions are updated, the configuration algorithm then calculates the fitness value of each particle. The fitness function considers 2 factors of deployment: item overlapping volume and the center-of-gravity of the box.

Item overlapping volume is the most important factor of deployment. Checking it ensures that each item can be packed into the box successfully. The system calculates the overlapping volume for each pair of items, and sums all of these overlapping volumes to determine the overlapping fitness value.

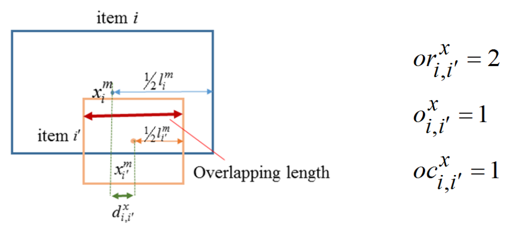
For items i and i', there are three potential overlapping relations in any one dimension: no overlapping relation, overlapping relation, and containing relation, as shown in Fig. 7.



(a). No overlapping relation: These two items have no overlapping areas.



(b). Overlapping relation: These two items have partial overlap.



(c). Containing relation: One item is contained in the other.

FIGURE 7. Three overlapping relations in the x-dimension for items *i* and *i'*. (a). No overlapping relation: These two items have no overlapping areas. (b). Overlapping relation: These two items have partial overlap. (c). Containing relation: One item is contained in the other.

We examine two conditions to distinguish between these relations. Take the x-dimension as an example; the distance between the two item centers is defined as $d_{i,i'}^x$, and the half-length of items i and i' are $1/2l_i^m$ and $1/2l_{i'}^m$, respectively. When $d_{i,i'}^x$ is bigger than $1/2l_i^m$ plus $1/2l_{i'}^m$, the two items are not overlapping, as in Fig. 7(a). Otherwise, the two items have overlap. To express this, we define a binary variable that represents whether two items overlap or not with the formula:

$$o_{i,i'}^x = \begin{cases} 0, & \text{if } d_{i,i'}^x \geq 1/2(l_i^m - l_{i'}^m) \\ 1, & \text{otherwise} \end{cases} \tag{15}$$

To calculate the overlapping length appropriately, the system further examines how the two items overlap by:

$$oc_{i,i'}^x = \begin{cases} 0, & \text{if } d_{i,i'}^x > \left|1/2(l_i^m - l_{i'}^m)\right| \\ 1, & \text{otherwise} \end{cases} \tag{16}$$

When one item is contained in the other one, as in Fig 7. (c), the distance between the two centers will be smaller than the discrepancy between the half-lengths of the two items. Thus, the overlapping length is defined as the length of the smaller item. Otherwise, the items have a partial overlap, as shown in Fig 7. (b). The overlapping relations and overlapping length between the two items can be defined, respectively, as:

$$or_{i,i'}^x = o_{i,i'}^x + oc_{i,i'}^x \tag{17}$$

$$l_{i,i'}^x = \begin{cases} 0, & \text{if } or_{i,i'}^x = 0 \\ 1/2(l_i^m + l_{i'}^m) - d_{i,i'}^x, & \text{if } or_{i,i'}^x = 1 \\ l_i^m \text{ or } l_{i'}^m, & \text{if } or_{i,i'}^x = 2 \end{cases} \tag{18}$$

where $or_{i,i'}^x$ represents the overlapping relation between the two items and $l_{i,i'}^x$ is the overlapping length. If the items have no overlapping relation, $or_{i,i'}^x = 0$, the two items do not overlap in the x-dimension, and the overlapping length is 0. If they have an overlapping relation, the two items have overlap and $o_{i,i'}^x = 1$. If neither item is fully contained by the other, $oc_{i,i'}^x = 0$, so the overlapping relation, $or_{i,i'}^x$, is 1. If the items have a containing relation, one item is contained in the other, so both $o_{i,i'}^x$ and $oc_{i,i'}^x$ are 1, and $or_{i,i'}^x$ is 2.

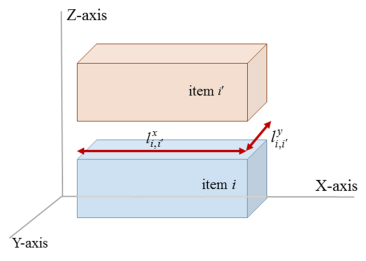
The overlapping relations are calculated separately for each of the three dimensions, and the results are combined to determine whether the two items overlap each other in 3D space:

$$o_{i,i'} = o_{i,i'}^x \cdot o_{i,i'}^y \cdot o_{i,i'}^z \tag{19}$$

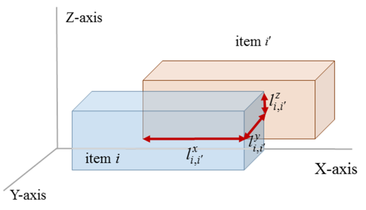
Only when the two items overlap in all three dimensions are they overlapping in 3D space. Some examples are demonstrated in Fig. 8. The overlapping volume can be calculated by:

$$v_{i,i'} = \begin{cases} 0, & \text{if } o_{i,i'} = 0 \\ l_{i,i'}^x \cdot l_{i,i'}^y \cdot l_{i,i'}^z, & \text{otherwise} \end{cases} \tag{20}$$

When $o_{i,i'}$ is 0, the two items have no overlap and the overlapping volume is 0. Otherwise, the volume of the overlapping



(a)



(b)

FIGURE 8. Examples of overlapping situations in 3D space for items *i* and *i'*. (a). These two items have overlapping lengths in the x- and y-dimensions, but not in the z-dimension. Thus, these two pieces of merchandise have no overlap in 3D space. (b). These two items have overlapping lengths in all three dimensions, so they are overlapping in 3D space.

cuboid is the product of the overlapping lengths in the three dimensions. After all these calculations are completed, the overlapping volumes from every pair of items are summed to give the total overlapping volume:

$$V = \sum_{i=1}^{I} \sum_{i'=i+1}^{I} v_{i,i'} \tag{21}$$

The fitness function of the item overlapping volume is defined as:

$$F_{overlap} = \frac{1}{V+\varepsilon} \tag{22}$$

where ε is a very small positive number that is added to the denominator to avoid a mathematical error.

The other important factor in item deployment is the position of the center-of-gravity (COG) of the box. Ideally, the COG will be close to the bottom center of the box, because this makes the box more stable and easier to carry.

The COG position is defined as $G = (\bar{x}, \bar{y}, \bar{z})$, and the central bottom position of the box is $C_j = (x^b_{cj}, y^b_{cj}, z^b_{cj})$. We calculate the COG by:

$$\begin{aligned} \bar{x} &= \sum_i^I w_i x_i^m / \sum_i^I w_i \\ \bar{y} &= \sum_i^I w_i y_i^m / \sum_i^I w_i \\ \bar{z} &= \sum_i^I w_i z_i^m / \sum_i^I w_i \end{aligned} \tag{23}$$

where w_i is the weight of the item m_i and (x_i^m, y_i^m, z_i^m) is the center position of the item m_i. The shorter the distance between the COG and the central bottom position of the box, the better, and this distance is defined as:

$$d^{COG} = \|G - C\| \tag{24}$$

The COG fitness function is defined as:

$$F_{COG} = 1/d^{COG} \tag{25}$$

Therefore, the fitness function of the configuration algorithm can be defined as:

$$F = \omega_1 F_{overlap} + \omega_2 F_{COG} \tag{26}$$

where ω_1 and ω_2 are weighted values determined by the experiments. When the configuration algorithm reaches its final iteration, the system will stop and test to see if the items are overlapping or not. If the items do not overlap with each other, the arrangement is a feasible solution.

Ideally, each combination would be run through the above adaptive PSO-based configuration algorithm to find all feasible solutions and to choose the optimal one. However, the number of potential combinations becomes more and more enormous with increasing numbers of items. Searching for the global optimal solution takes a lot of computation time and is not suitable for a real-time automated packaging system.

Therefore, we use two searching strategies to decide on the final solution. In the first searching strategy, the algorithm stops when the first feasible solution is found; this solution will be the chosen one. In the second searching strategy, the algorithm stops when a predetermined number of feasible solutions are found, and chooses the best one among them by comparing their fitness values.

The system then generates a packing sequence, based on the chosen combination, and translates the deployment results and the packing sequence to each machine and robot in the distribution process through the IoT-based cyber network.

B. THREE DIVIDING STRATEGIES

When all the items in an order cannot be packed into a single box, the system will divide the items into two groups. Based on the need for packaging to be convenient [53], the ideal is that all boxes belonging to the same order have a similar size. Based on this principle, the total volumes of each group of items should be similar. To achieve this, we propose three dividing strategies.

1) DIVIDING STRATEGY I

This is a simple strategy in which all items are ranked by their volumes and are divided into two groups according to rank: odd numbers in one group and even numbers in another group. Afterwards, each group runs the adaptive PSO-based packing algorithm separately.

2) DIVIDING STRATEGY II

In this strategy, all the items are first ranked by volume and the total volume of all the items is also calculated. The system works from the smallest item to the largest, adding the volume of each new item to an accumulating result. When this accumulated volume exceeds half of the total volume, the included items will be grouped together, while the remaining items are assigned to the second group.

3) DIVIDING STRATEGY III

This strategy is similar to strategy II. Both strategies pick items and accumulate their volumes until the accumulated volume exceeds half of the total volume. The major difference is the sequence for picking the items. Strategy III alternately picks the biggest item and the smallest item from those unselected, working towards the middle until the accumulated volume exceeds half of the total. In other words, the strategy groups the largest and smallest items together, while the remaining merchandise makes up the second group.

All of these dividing strategies divide all items into two groups, and each group runs the adaptive PSO-based packing algorithm to choose a suitably sized box and the deployment of items. When a group of items cannot be packed into a single box, the system will again divide these items, forming two sub-groups by the same dividing strategy.

TABLE 1. Examples of Merchandise

Item	Toilet paper holder	Life jacket for dogs	Lotion
Picture			
Weight(g)	848	200	259
Size(cm)	11×35×10.2	7×21×28	22×9×4.5
Item	Can	Dried bonito flakes	Single Lens Reflex Camera
Picture			
Weight(g)	240	181	1300
Size(cm)	7×6.8×6.8	4.6×29.4×22	25.1×14.4 ×10.5
Item	Sphygmomanometer	Umbrella	Clock
Picture			
Weight(g)	850	95	82
Size(cm)	14.5×16.3×15.4	32×12×1.5	3.8×3×3

V. SIMULATIONS

In this section, simulations of the 3D adaptive PSO-based packing algorithm are presented to demonstrate the effectiveness of the proposed method. We chose 50 pieces of merchandise from the Amazon Japan website [54], including daily necessities, cameras, instant foods, kitchen utensils, toys, pet supplies, and computer peripherals. Some examples are shown in Table 1. The size and shape of the items were strongly heterogeneous; the size of the largest item was 21.1cm × 19.8cm × 20.6cm and the size of the smallest was 3.8cm × 3cm × 3cm. Based on volume, the merchandise was divided into large items and small items, with the separation threshold at 1000cm^3. The sizes of the boxes used in the simulations were the same as the boxes used by DHL International Express [55], as tabulated in Table 2.

TABLE 2. Size of the boxes.

Box Number	Recommended Max Weight(kg)	Length(cm)	Width(cm)	Height(cm)
Box 2	1	33.7	18.2	10.0
Box 3	2	33.7	32.2	10.0
Box 4	5	33.7	32.2	18.0
Box 5	10	33.7	32.2	34.5
Box 6	15	41.7	35.9	36.9
Box 7	20	48.1	40.4	38.9
Box 8	25	54.1	44.4	40.9

In the first experiment, we defined three different scenarios and generated six orders for each scenario. We tested each order three times to compare the COG distances and the computation times of the two searching strategies. In all cases, the maximum number of iterations for each combination is 40, and the number of feasible solutions in searching strategy II is 10. In every case, the number of items chosen was between four and six, so all the items could be packed inside just one box. The items in the first scenario were all large items and the items in the second scenario were all small. The third scenario included both large items and small items. The results for the three scenarios are listed in Table 3 to Table 5.

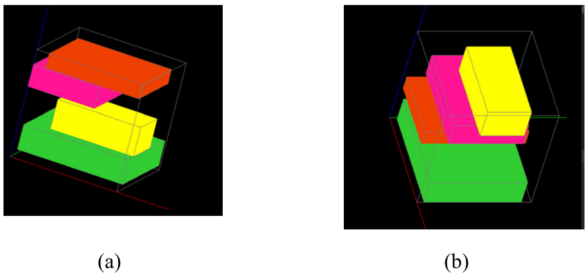


FIGURE 9. Item deployments for order A5, according to the two strategies. (a). Searching Strategy I (b). Searching Strategy II.

A box's filling rate is calculated by dividing the total volume of items by the volume of the box; thus, it is determined in the process of searching for an appropriate box size and is not influenced by the searching strategy used. The results of the first scenario are shown in Table 3; the filling rates across all the orders were stable and the average rate was around 45%. This shows that the algorithm can successfully choose an appropriate size of box to contain all the necessary items.

Compared with only searching for the first feasible solution, searching for ten feasible solutions and choosing the best one brought the center of gravity closer to the central bottom area. Therefore, the COG distances, d^{COG}, for searching strategy II are significantly shorter than for searching strategy I.

Of course, searching for more solutions takes more time; it results in strategy II taking 2.2 times as long as strategy I for computation. However, the computation time is highly relative depending on the specific items in an order. In some cases, feasible solutions are easily found, so neither strategy needs to spend much time searching, as demonstrated in order A5. In other cases, there is no feasible solution for the first chosen box (5) so the algorithm must upgrade the box size and recalculate all combinations. Since every combination is calculated for the first chosen box, more time has to be spent and both strategies need a longer computation time, as demonstrated in orders A1, A3, A4, and A6. In other cases, if feasible solutions are hard to find, searching for multiple feasible solutions will take much more time, resulting in a much greater computation time for searching strategy II than searching strategy I, as in order A2. Fig. 9 is an example of the differing deployment of items between the two strategies. The deployment result from strategy II looks better than the other.

Table 4 lists the results from the second scenario, in which all the items were small. The filling rates were variable but the average filling rate was lower than for the first scenario. However, the smallest box was chosen in each of the

TABLE 3. Comparison of the two searching strategies for large items.

				Searching Strategy I		Searching Strategy II	
Order	Large Items	Box Number	Filling Rate	d^{COG} (cm)	Computation Time(s)	d^{COG} (cm)	Computation Time(s)
A1	29, 8, 20, 47, 31, 5	5	44.11	19.3 (1.24)	156 (6.89)	16.6 (0.43)	187 (9.27)
A2	43, 18, 5, 9, 9, 9	8	39.77	20.5 (5.05)	0.17 (0.17)	16.5 (1.59)	1.38 (0.71)
A3	41, 41, 46, 46, 44	5	42.46	14.5 (1.00)	5.67 (0.27)	13.8 (0.93)	5.90 (0.12)
A4	7, 24, 24, 45, 8	5	42.18	15.9 (1.47)	37.9 (0.95)	13.7 (1.27)	38.0 (0.47)
A5	27, 7, 18, 46	5	46.54	12.3 (0.09)	0.55 (0.02)	12.3 (0.16)	0.76 (0.02)
A6	31, 31, 40, 47	4	54.69	10.3 (1.73)	8.05 (0.22)	9.2 (1.60)	8.60 (0.17)
Average			44.96	15.50 (1.76)	34.73 (1.41)	13.71 (0.99)	40.42 (1.79)

Note: *w* is 1.

TABLE 4. Comparison of the two searching strategies for small items.

				Searching Strategy I		Searching Strategy II	
Order	Small Items	Box Number	Filling Rate (%)	d^{COG} (cm)	Computation Time(s)	d^{COG} (cm)	Computation Time(s)
B1	49, 12, 26, 23, 35,16	2	46.06	6.58 (0.51)	0.49 (0.03)	6.22 (1.17)	8.49 (7.40)
B2	16, 16, 25, 34, 3, 50	2	44.48	5.44 (0.25)	0.07 (0.01)	5.00 (0.26)	0.59 (0.02)
B3	48, 10, 13, 11, 2	2	40.06	6.47 (1.33)	0.17 (0.13)	5.47 (1.15)	1.29 (0.67)
B4	21, 21, 33, 33, 33	2	37.86	5.80 (1.85)	0.06 (0.00)	4.99 (0.65)	0.39 (0.01)
B5	26, 26, 25, 11	2	24.86	6.46 (0.86)	0.05 (0.01)	3.87 (0.26)	0.29 (0.01)
B6	1, 14, 14, 33	2	23.37	4.80 (0.67)	0.03 (0.00)	4.46 (0.37)	0.30 (0.01)
Average			32.25	5.92 (0.91)	0.15 (0.03)	5.00 (0.64)	1.89 (1.35)

Note: *w* is 1.

TABLE 5. Comparison of the two searching strategies for mixed large and small items.

					Searching Strategy I		Searching Strategy II	
Order	Large Items	Small Items	Box Number	Filling Rate (%)	d^{COG} (cm)	Computation Time(s)	d^{COG} (cm)	Computation Time(s)
C1	43, 31	26, 26, 33, 33	4	44.27%	10.92 (1.54)	0.09 (0.03)	9.93 (1.59)	4.29 (1.81)
C2	29, 44, 38	50, 13, 1	5	17.45%	17.40 (3.72)	0.24 (0.27)	13.63 (0.63)	0.54 (0.01)
C3	8, 31, 31, 44	50	4	54.28%	9.69 (0.79)	4.49 (1.16)	8.90 (1.15)	9.10 (3.03)
C4	17, 44	48, 48, 3	3	54.30%	7.46 (1.22)	0.31 (0.15)	4.87 (1.01)	2.66 (0.01)
C5	6, 6, 24	42	6	21.24%	14.25 (0.26)	0.05 (0.00)	13.78 (0.47)	0.28 (0.00)
C6	17, 17	11, 14	3	66.93%	5.91 (0.96)	0.21 (0.01)	5.67 (0.61)	2.32 (0.07)
Average				43.08%	10.94 (1.41)	0.90 (0.27)	9.46 (0.91)	3.20 (0.82)

Note: *w* is 1.

six orders, so the filling rates were dictated by the total volume of the items. Since the first chosen box was also the final box in every case, the computation times of orders, for the most part, were very short. In searching strategy I, all orders could be calculated in less than 1 second; in searching strategy II, more than half of the orders could be done in 1 second. Comparing the two strategies, searching for multiple solutions shortened the COG distance by 15%, a statistically significant improvement with a p-value of 0.023 in a t-test. The deployment of items in order B1 is shown in Fig. 10.

The orders in the third scenario included both large items and small items; the results are tabulated in Table 5. Across the six orders, the filling rates ranged between 17% and 66% and the average filling rate was 43%. The huge range in filling rate is due to multiple reasons. When one of the large items is very big and requires a larger sized box to contain it, the filling rate is pulled down, as shown in Fig. 11(a). Otherwise, when the small items just fit between the large items, it results in a higher filling rate, as shown in Fig. 11(b).

A summary of COG distances for these three scenarios is given in Table 6. Generally, searching for multiple

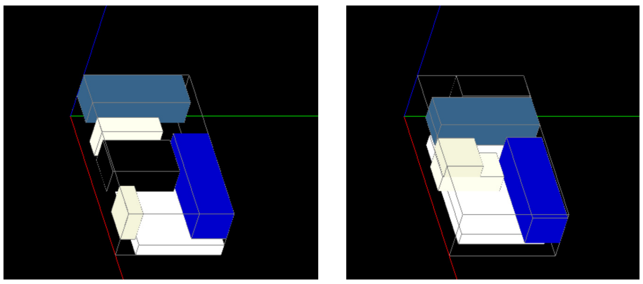

(a) (b)

FIGURE 10. Item deployments for order B1, according to the two strategies. (a). Searching Strategy I (b). Searching Strategy II.

solutions and choosing the best one can statistically significantly shorten the distance between the COG and the bottom center of the box. This means that the items are closer to the center and bottom of the box, and the heaviest items are placed close to the bottom of the box. In all three scenarios, the COG distances of searching strategy II are significantly shorter than those of searching strategy I. The results of the t-test are also listed in Table 6.

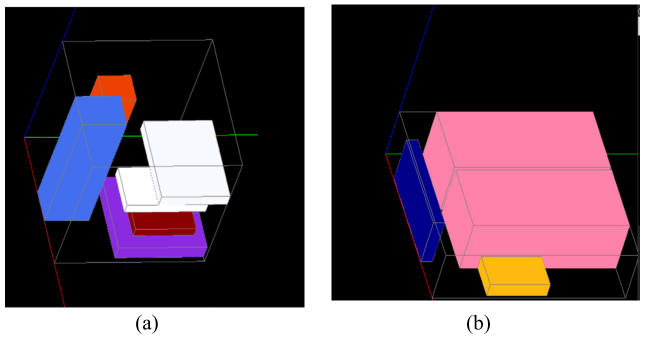


FIGURE 11. Examples of item deployments in scenario C. (a). Order C2 (b). Order C6.

TABLE 6. T-test results for COG distances.

	Mean(SD)			
	SS-I	SS-II	Improved Rate	p-value
Scenario A	15.5 (1.76)	13.7 (0.99)	10.5%	0.015 *
Scenario B	5.92 (0.91)	5.00 (0.64)	15.5%	0.023*
Scenario C	10.9 (1.41)	9.46 (0.91)	13.4%	0.024*

Note: SS-I and SS-II are abbreviations for Searching Strategy I and Searching Strategy II, respectively.
* $p < 0.05$.

TABLE 7. T-test results for computation times.

	Mean(SD)			
	SS-I	SS-II	Magnification	p-value
Scenario A	34.7 (1.41)	40.4 (1.79)	1.638	0.163
Scenario B	0.15 (0.03)	1.89 (1.35)	12.90	0.112
Scenario C	0.90 (0.27)	3.20 (0.82)	3.557	0.014*

Note: SS-I and SS-II are abbreviations for Searching Strategy I and Searching Strategy II, respectively.
* $p < 0.05$.

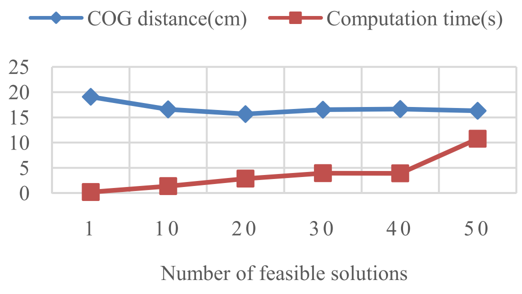


FIGURE 12. COG distance and computation time of different number of feasible solutions.

As the computation time is highly influenced by how difficult the feasible solutions are to find, there are big differences between the orders. Table 7 compares the computation times between the two searching strategies. In general, searching for more feasible solutions undoubtedly takes more time, so how many feasible solutions should be found? To answer this, we must more carefully examine the number of feasible solutions and compare the COG distances and computation times. We chose the order A2 as an example, and ran it three times for different number of feasible solutions; the results are shown in Fig. 12. It show that the COG distances become shorter as the number of feasible solutions increase. Similarly, the computation time increases as the number of feasible solutions increase. However, the improvement in COG distance is not stable and it is hard to decide which number of feasible solutions is best. On the contrary, the increase in computation time is directly proportional to the number of feasible solutions found. For a real-time automated packaging system, the computation time needs to be as short as possible, so we chose 10 as the number of feasible solutions for searching strategy II.

In the second experiment, we examined the performance of the three dividing strategies by generating 9 orders, containing both large items and small items, where all the items could not fit inside a single box. Therefore, the items had to be separated into two groups and the packing algorithm was adopted separately for each group. How the items are divided into two groups impacts the box choices. Considering customers' convenience, we intend for the two boxes to have similar sizes and similar filling rates.

Table 8 lists the performances of the three dividing strategies. We can compare these three strategies according to average filling rate, discrepancy in filling rate, and discrepancy in box size. Average filling rate and discrepancy in filling rate are the sum and difference of the filling rates of the two boxes, respectively. The discrepancy of box size is the difference in box number between the two boxes. A small discrepancy value means the two boxes were similar in size.

Dividing strategy I separates the items by odd number and even number, so equal numbers of large and small items are found in each group. Of the three strategies, this one had the worst filling rate but resulted in the smallest discrepancy in box size. In dividing strategy II, the items are ranked by volume and are totted up until the accumulated volume exceeds half of the total volume. Therefore, one group contains larger items while the other group contains smaller items. For the smaller items group, the box size was reduced and the filling rate was increased. This method resulted in the highest average filling rate but the largest discrepancies in both filling rate and box size. Dividing strategy III puts the largest items and the smallest items in one group, and the medium-sized items in another. The performance of this strategy on all three metrics was in the middle, compared to the other methods. Compared to dividing strategy I, the bigger box in dividing strategy III generally contained more large items, meaning that the remaining items could be put inside a smaller box. This meant that the filling rate was higher, but the discrepancy in box size was also higher. One example of item deployments is shown in Fig. 13. Generally speaking, each strategy had its own advantages and disadvantages; however, each of the three dividing strategies can separate items into two groups reasonably well.

We further demonstrate a special case that mimics the common type of package in which all the items are identical. In this case, there are multiple identical small items in a single order; the deployment is shown in Fig. 14. The results show that the packing algorithm can not only be used for strongly heterogeneous items, but also for identical items.

TABLE 8. Comparison of the three dividing strategies.

		Average filling rate			Discrepancy in Filling Rate			Discrepancy in Box Size		
Order	Number of items	DS-I	DS-II	DS-III	DS-I	DS-II	DS-III	DS-I	DS-II	DS-III
D1	15	42.47	40.51	33.75	12.48	11.67	1.36	2	3	1
D2	15	36.20	40.76	40.67	2.52	15.73	14.52	1	1	1
D3	15	36.60	42.59	36.76	8.22	7.75	10.22	1	0	1
D4	13	44.95	44.95	44.95	9.97	10.49	14.75	0	0	0
D5	13	27.73	37.45	37.06	2.22	13.91	12.42	0	2	2
D6	13	29.96	34.22	41.14	5.38	5.35	19.85	0	1	2
D7	11	40.84	39.90	40.99	11.80	2.06	13.38	1	1	1
D8	11	37.29	47.88	37.29	6.50	35.67	0.97	0	1	0
D9	11	27.62	28.47	27.77	20.56	24.34	21.23	3	3	3
Average		35.96	39.64	37.82	8.85	14.11	12.08	0.89	1.33	1.22

Note: *w* is 2, DS-I, DS-II, and DS-III are abbreviations for Dividing Strategies I to III, respectively.

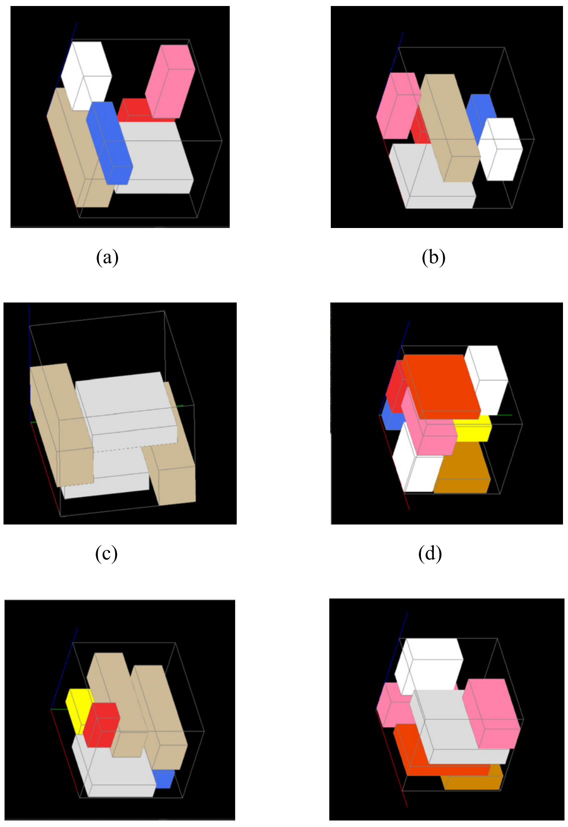


FIGURE 13. Item deployments of order D5, according to the three dividing strategies. The order contains 13 items and the sequence of ranked items is 9, 9, 5, 5, 41, 43, 17, 17, 18, 29, 45, 8. (a). Box one of DS-I (item numbers: 9, 5, 41, 43, 18, 45) (b). Box two of DS-I (item numbers: 9, 5, 43, 17, 29, 8). (c). Box one of DS-II (item numbers: 9, 9, 5, 5). (d). Box two of DS-II (item numbers: 41, 43, 43, 17, 18, 29, 45, 8). (e). Box one of DS-III (item numbers: 9, 9, 45, 8, 5, 29). (f). Box two of DS-III (item numbers: 5, 41, 43, 17, 17, 18).

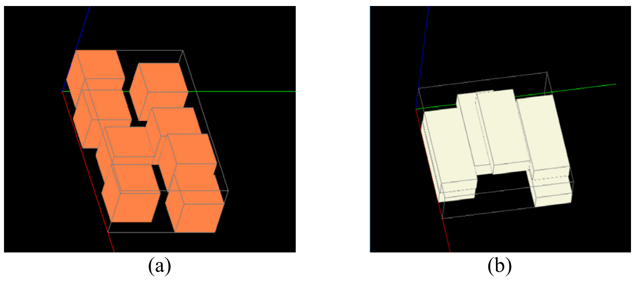


FIGURE 14. Item deployments of identical items. (a). Cube item. The item size is 7×6.8×6.8cm, the box number is 2, and the filling rate is 42.22%. (b). Cuboid item. The item size is 22×9×4.5cm, the box number is 3, and filling rate is 49.26%.

TABLE 9. Filling rate of real packages.

Order	Box	Amount of items	Filling rate
1	box 1	4	52.23
2	box 1	4	36.29
3	box 1	4	47.69
	box 2	1	38.78
4	box 1	4	32.19
	box 2	2	11.44
Average			36.43

Finally, we compared our algorithm results with real world packages. We placed 4 orders on the popular B2C online shopping website in Taiwan [56] and received 6 boxes. The details of the real orders were similar to simulation scenario C in the first experiment, in that each order contained large items and small items and the amount of items was between four and six. The results are tabulated in Table 9 and some pictures are shown in Fig. 15. The filling rates were between 11% and 52% and the average filling rate was 36%. We can treat the real packaging filling rate as a base line, and find out that the packages packed by our algorithm has a competitive performance. We also found that some bubble wrap were placed to protect the merchandises in the real packages. Since we have leaved some buffer space in the box and calculated the total volume of items. The suitable volume of cushioning material can be calculated and the bubble wrap can be placed between the items and the box.

The simulations in this section show that the proposed 3D adaptive PSO-based packing algorithm can solve the difficult 3D multiple bin size bin packing problem, with a strong heterogeneity of items and variable sizes of box, in a wide range of scenarios. Furthermore, our packing algorithm produces competitive packed boxes with the real packages we assessed. The next section will further demonstrate the practicability of

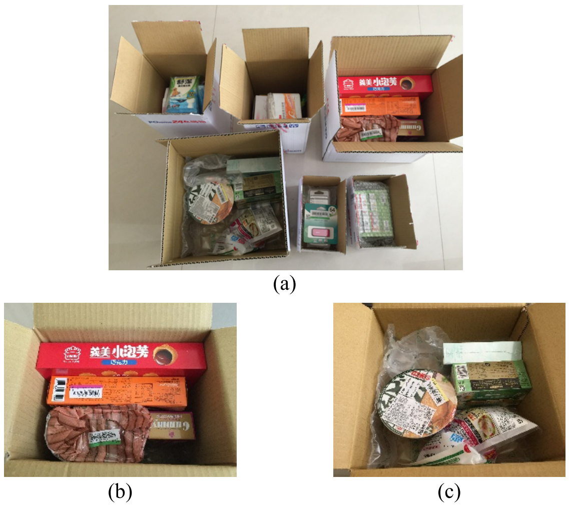


FIGURE 15. Pictures of the item deployment. (a). The picture of all packages. (b). The picture of order R1. (c). The picture of box 1 of order R4.

the packing algorithm by implementing it in the automated e-fulfillment packaging system.

VI. IMPLEMENTATION AND EXPERIMENTS

To demonstrate the IoT-based automated e-fulfillment packaging system, we implemented a packaging scenario in our laboratory which consisted of a mobile application (app), a server, a delivery robot, and a robot manipulator. The server connects all devices by constructing a cyber network through the User Datagram Protocol (UDP), which can receive and send a lot of data to multiple devices [56]. In this way, the network can send different data to both the delivery robot and the robot manipulator.

When a customer places an order on her/his smart device, such as a smart phone, through an e-shopping app, the server receives the order and runs the adaptive PSO-based packing algorithm to decide upon a suitably sized box as well as the arrangement for items in the box. The calculated results are then sent to the delivery robot as well as the robot manipulator. Following this, the robot manipulator grasps the box and puts it on the work table (in this implementation, we used a work table to replace a conveyor), while the delivery robot grasps items from the preparation table and places them on the work table. Then the robot manipulator packs the items into the box. Fig.16 illustrates this demonstration scenario. The robots keep communicating with each other about their execution status to avoid collisions and to ensure that the packaging task goes smoothly.

The e-shopping app is constructed with the PHP and HTML website languages and provides merchandise information to customers and translates placed orders to the server. The delivery robot, designed and implemented by our laboratory, is equipped with a vision module, a speech module, a laser module, two 6-DOF arms, and a moving platform. The robot can recognize objects through a real-time object recognition system, which combines a CUDA SURF detector and a BRISK descriptor to improve the computing speed using a low computation load [57]. The robot can recognize the items of the order currently executing among all the items on the preparation table, regardless of whether it knows the placement of the items.

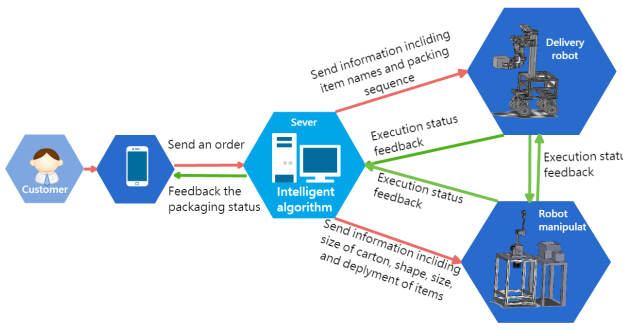

Customer
Send an order
Feedback the packaging status
Sever
Intelligent algorithm
Send information inclidng item names and packing sequence
Delivery robot
Execution status feedback
Execution status feedback
Execution status feedback
Robot manipulat
Send information inclidng size of carton, shape, size, and deplyment of items

FIGURE 16. Scenario of intelligent packaging task.

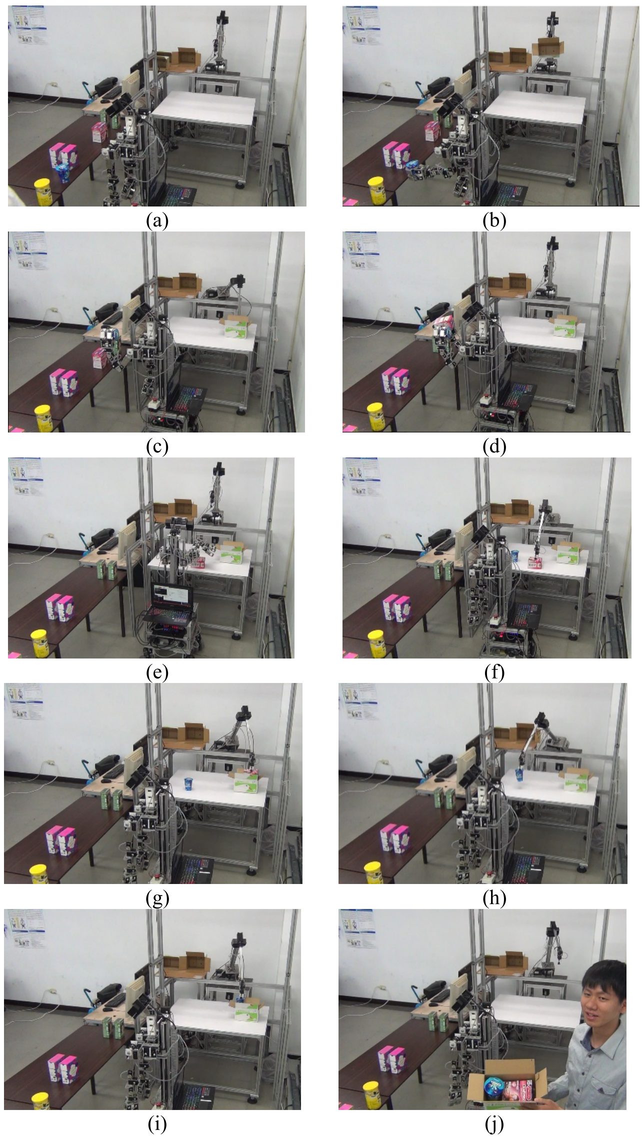
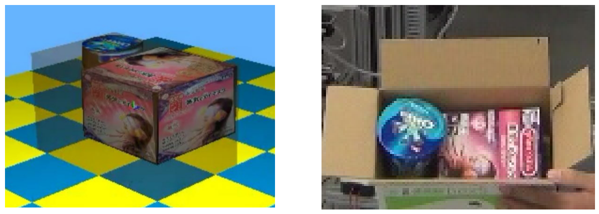


FIGURE 17. Snapshots of the execution process of experiment I.



(a) (b)

FIGURE 18. The comparison of the planned deployment and the executed one.(The filling rate is 37.54).

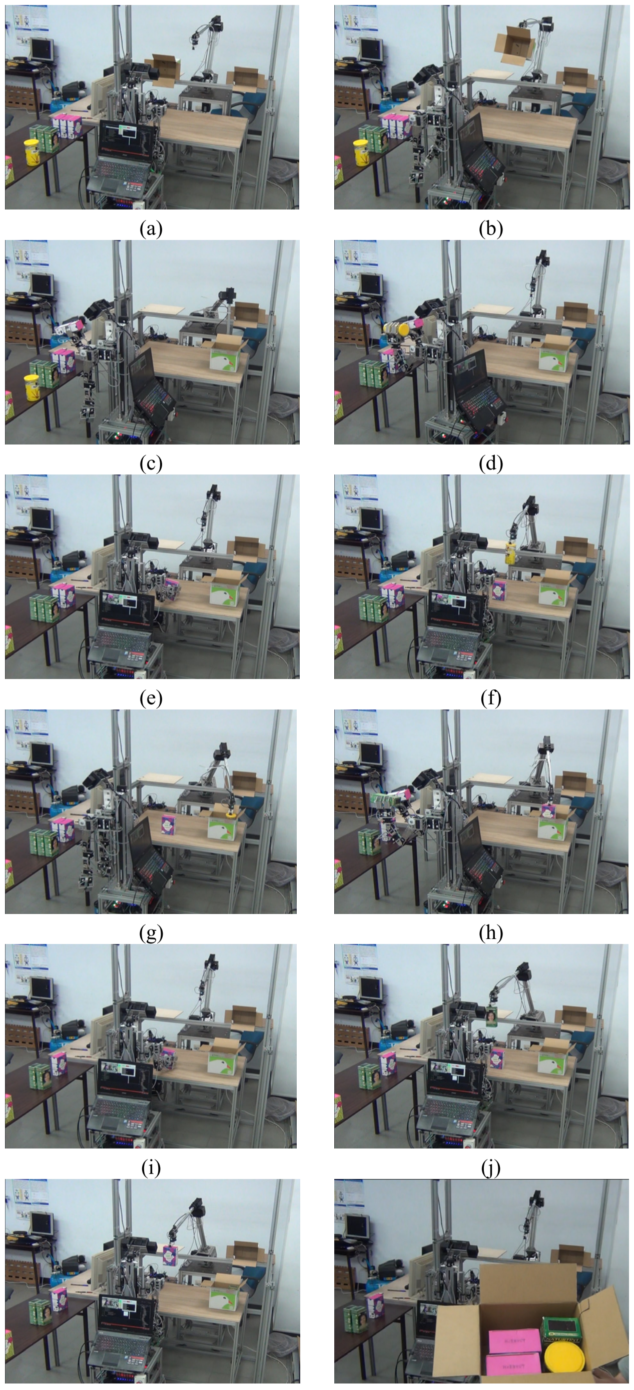
The robot manipulator has 6 degrees of freedom and is equipped with an eye-to-hand visual servo system for recognizing and grasping objects. Its 3D working space is around 150cm×150cm×70cm. The packing algorithm used in this experiment was a simplified vision and the number of box size and merchandise were fewer than the simulations. There were only two sizes of box (small - 23cm×14cm×13.5cm, and large - 23cm×18.5cm×19.5cm) and 6 kinds of merchandise used in the experiments.

In order to evaluate the performance of the proposed system, we examined two orders with different quantities and types of merchandise. In the first experiment, we ordered an eye mask and an Oreo cookie. Fig. 17 shows snapshots of the packaging process. When the item arrangement had been calculated by the packing algorithm, the results were sent to the delivery robot and the robot manipulator. The robot manipulator first picked a small box and put it on the work table, as in Fig. 17(b)-(e), while the delivery robot grasped the appointed items and put them on the work table. Once the first batch of items had been put on the table, the delivery robot informed the robot manipulator.

The robot manipulator then placed the items into the box according to the arrangement calculated by the configuration algorithm as in Fig. 17(f)-(i). By comparing the planned deployment (Fig. 18 (a)) with the executed one (Fig. 18 (b)), the experiment demonstrates both the practicability of the packing algorithm and the efficiency of the proposed system. Both the delivery robot and the robot manipulator can successfully execute the order sent from the server.

In the second experiment, we tested a more complex order, consisting of a hair colorant, two firming masks, and a wafer roll. Fig. 19 shows snapshots of the packaging process. This time, to pack all the items, the robot manipulator chose a large box. The experiments also demonstrated that the robots in the network can cooperatively accomplish a task by keeping in touch with each other about their executing status. For example, the delivery robot put the second batch of items on the work table only after the robot manipulator had already packed the first batch of items into their box. This ensures that neither the robots nor the items will crash into each other.

These two experiments illustrate the efficiency of the proposed system. Based on the concepts of Industry 4.0, we implemented an IoT-based automated e-fulfillment packaging system to deal with orders sent from users online.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l)

FIGURE 19. Snapshots of the execution process of experiment II.

Although we have only examined this system in a limited laboratory environment with a mobile robot and a robot manipulator, the experiments have shown the feasibility of implementing this system in a real smart factory. Through communicating and cooperating throughout the work, robots can complete the task safely without any collisions. Safety is the most basic and important issue when real implementation takes place in a smart factory, because any collision could cause huge losses for the enterprise.

VII. CONCLUSION

The warehouse order fulfillment task for e-shopping is different from the transitional retuning tasks in which machines do the same thing all the time. To deal with each order individually, all devices in the packaging system have to work as a team. In this team, each device acquires its working status, and sends feedback to the others.

To accomplish the packaging task, an IoT-based automated e-fulfillment packaging system has been proposed in this paper. The intelligent machines in this system are equipped with data sensing and acquisition abilities, local computing and processing units, and wireless transmitting modules. Meanwhile, the server integrates several intelligent algorithms to rearrange the schedule of the orders, to calculate the deployment of items in a box, and to monitor the machines.

Due to the huge diversity between orders, choosing a suitably sized box and generating an arrangement of items in the box are difficult problems. This paper has proposed a 3D adaptive PSO-based packing algorithm to deal with them. It compares the size and volume of items and boxes to select a suitably sized box, and utilizes an adaptive PSO-based configuration algorithm to generate the arrangement of items. When all items cannot be packed into a single box, the algorithm will divide the items into two groups according to one of three dividing strategies.

We conducted several simulations to examine the performance of the proposed packing algorithm. The first simulation compared the COG distances and the computation times of two searching strategies in three different scenarios. The results showed that searching for more feasible solutions significantly shortened the COG distances but required more computation time. The second simulation compared the performance of the three dividing strategies according to their filling rates and the diversity of box sizes. The results showed that each strategy has its advantages and disadvantages. The simulation results were also compared with the filling rates of real packages, and the results showed that the proposed algorithm has better performance.

Finally, we implemented and examined the proposed system using a mobile application (app), a server, a delivery robot, and a robot manipulator. The results of the experiments have demonstrated the practicability of the packing algorithm and the efficiency of the system, in which all devices communicate with each other well and the delivery robot and robot manipulator pack items in a cooperative and smooth manner.

REFERENCES

[1] N. K. Jain, H. Gajjar, B. J. Shah, and A. Sadh, "A conceptual framework for measuring e-fulfillment dimensions: A consumer perspective," *J. Internet Commerce*, vol. 14, no. 3, pp. 363–383, 2015.

[2] J. M. Tarn, M. A. Razi, H. J. Wen, and A. A. Perez, Jr., "E-fulfillment: The strategy and operational requirements," *Logistics Inf. Manage.*, vol. 16, no. 5, pp. 350–362, 2003.

[3] N. A. H. Agatz, M. Fleischmann, and J. A. E. E. Van Nunen, "E-fulfillment and multi-channel distribution—A review," *Eur. J. Oper. Res.*, vol. 187, no. 2, pp. 339–356, 2008.

[4] R. D'Andrea, "Guest editorial: A revolution in the warehouse: A retrospective on kiva systems and the grand challenges ahead," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 4, pp. 638–639, Oct. 2012.

[5] (2015). *Automation is Key in E-Commerce Fulfillment, Swisslog White Paper*, accessed on Jul. 15, 2016. [Online]. Available: http://www.swisslog.com

[6] C. Liang *et al.*, "Automated robot picking system for e-commerce fulfillment warehouse application," in *Proc. Int. Fed. Promotion Mech. Mach. Sci. World Congr.*, 2015, doi: 10.6567/IFToMM.14TH.WC.OS13.077.

[7] H. Zhang *et al.* (2016). "DoraPicker: An autonomous picking system for general objects." [Online]. Available: https://arxiv.org/abs/1603.06317

[8] R. D'Andrea, "Guest editorial can drones deliver?" *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 647–648, Jul. 2014.

[9] J. Wu, M. Dong, K. Ota, M. Tariq, and L. Guo, "Cross-domain fine-grained data usage control service for industrial wireless sensor networks," *IEEE Access*, vol. 3, pp. 2939–2949, 2015.

[10] J. Wu, M. Dong, K. Ota, Z. Zhou, and B. Duan, "Towards fault-tolerant fine-grained data access control for smart grid," *Wireless Pers. Commun.*, vol. 75, no. 3, pp. 1787–1808, Apr. 2014.

[11] (2014). W. MacDougall, *Industrie 4.0 Smart Manufacturing for the Future*, accessed on Jul. 15, 2016. [Online]. Available: http://www.gtai.de/GTAI/Content/CN/Invest/_SharedDocs/Downloads/GTAI/Brochures/Industries/industrie4.0-smart-manufacturing-for-the-future-en.pdf

[12] C. Perera, C. H. Liu, S. Jayawardena, and M. Chen, "A survey on Internet of Things from industrial market perspective," *IEEE Access*, vol. 2, pp. 1660–1679, Jan. 2014.

[13] A. A. F. Saldivar, Y. Li, W. N. Chen, Z.-H. Zhan, J. Zhang, and L. Y. Chen, "Industry 4.0 with cyber-physical integration: A design and manufacture perspective," in *Proc. IEEE Int. Conf. Autom. Comput.*, Sep. 2015, pp. 1–6.

[14] F. Shrouf, J. Ordieres, and G. Miragliotta, "Smart factories in industry 4.0: A review of the concept and of energy management approached in production based on the Internet of Things paradigm," in *Proc. IEEE Int. Conf. Ind. Eng. Eng. Manage.*, Dec. 2014, pp. 697–701.

[15] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manuf. Lett.*, vol. 3, pp. 18–23, Jan. 2015.

[16] R. Baheti and H. Gill, "Cyber-physical systems," *Impact Control Technol.*, vol. 12, pp. 161–166, Mar. 2011.

[17] K. Zhou, T. Liu, and L. Zhou, "Industry 4.0: Towards future industrial opportunities and challenges," in *Proc. Int. Conf. Fuzzy Syst. Knowl. Discovery*, Aug. 2015, pp. 2147–2152.

[18] P. Helo, M. Suorsa, Y. Hao, and P. Anussornnitisarn, "Toward a cloud-based manufacturing execution system for distributed manufacturing," *Comput. Ind.*, vol. 65, no. 4, pp. 646–656, 2014.

[19] J. A. Bennell and J. F. Oliveira, "A tutorial in irregular shape packing problems," *J. Oper. Res. Soc.*, vol. 60, no. 1, pp. 93–105, 2009.

[20] G. Wäscher, H. Haußner, and H. Schumann, "An improved typology of cutting and packing problems," *Eur. J. Oper. Res.*, vol. 183, no. 3, pp. 1109–1130, 2007.

[21] X. Zhao, J. A. Bennell, T. Bektaş, and K. Dowsland, "A comparative review of 3D container loading algorithms," *Int. Trans. Oper. Res.*, vol. 23, nos. 1–2, pp. 287–320, 2016.

[22] E. K. Burke, R. S. Hellier, G. Kendall, and G. Whitwell, "Irregular packing using the line and arc no-fit polygon," *Oper. Res.*, vol. 58, no. 4, pp. 948–970, 2010.

[23] E. G. Birgin and F. N. C. Sobral, "Minimizing the object dimensions in circle and sphere packing problems," *Comput. Oper. Res.*, vol. 35, no. 7, pp. 2357–2375, 2008.

[24] M. Hifi and R. M'hallah, "A literature review on circle and sphere packing problems: Models and methodologies," *Adv. Oper. Res.*, vol. 2009, Apr. 2009, doi: 10.1155/2009/150624.

[25] C. D. Maranas, C. A. Floudas, and P. M. Pardalos, "New results in the packing of equal circles in a square," *Discrete Math.*, vol. 142, nos. 1–3, pp. 287–293, 1995.

[26] A. Grosso, A. R. M. J. U. Jamali, M. Locatelli, and F. Schoen, "Solving the problem of packing equal and unequal circles in a circular container," *J. Global Optim.*, vol. 47, no. 1, pp. 63–81, 2010.

[27] A. Grosso, M. Locatelli, and F. Schoen, "A population-based approach for hard global optimization problems based on dissimilarity measures," *Math. Programm.*, vol. 110, no. 2, pp. 373–404, 2007.

[28] C. O. Lopez and J. E. Beasley, "A heuristic for the circle packing problem with a variety of containers," *Eur. J. Oper. Res.*, vol. 214, no. 3, pp. 512–525, 2011.

[29] K. Fleszar and C. Charalambous, "Average-weight-controlled bin-oriented heuristics for the one-dimensional bin-packing problem," *Eur. J. Oper. Res.*, vol. 210, no. 2, pp. 176–184, 2011.

[30] J. Kang and S. Park, "Algorithms for the variable sized bin packing problem," *Eur. J. Oper. Res.*, vol. 147, no. 2, pp. 365–372, 2003.
[31] A. Lodi, S. Martello, and M. Monaci, "Two-dimensional packing problems: A survey," *Eur. J. Oper. Res.*, vol. 141, no. 2, pp. 241–252, 2002.
[32] E. B. C. H. Hopper and B. C. Turton, "An empirical investigation of metaheuristic and heuristic algorithms for a 2D packing problem," *Eur. J. Oper. Res.*, vol. 128, no. 1, pp. 34–57, 2001.
[33] A. de Almeida and M. B. Figueiredo, "A particular approach for the three-dimensional packing problem with additional constraints," *Comput. Oper. Res.*, vol. 37, no. 11, pp. 1968–1976, 2010.
[34] C. H. Che, W. Huang, A. Lim, and W. Zhu, "The multiple container loading cost minimization problem," *Eur. J. Oper. Res.*, vol. 214, no. 3, pp. 501–511, 2011.
[35] M. Eley, "Solving container loading problems by block arrangement," *Eur. J. Oper. Res.*, vol. 141, no. 2, pp. 393–409, 2002.
[36] R. R. Amossen and D. Pisinger, "Multi-dimensional bin packing problems with guillotine constraints," *Comput. Oper. Res.*, vol. 37, no. 11, pp. 1999–2006, 2010.
[37] Y. Wu, W. Li, M. Goh, and R. de Souza, "Three-dimensional bin packing problem with variable bin height," *Eur. J. Oper. Res.*, vol. 202, no. 2, pp. 347–355, 2010.
[38] J. F. Gonçalves and M. G. C. Resende, "A biased random key genetic algorithm for 2D and 3D bin packing problems," *Int. J. Prod. Econ.*, vol. 145, no. 2, pp. 500–510, 2013.
[39] 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 *Proc. Portuguese Conf. Autom. Control*, 2015, pp. 355–364.
[40] W. Zhu, W. Huang, and A. Lim, "A prototype column generation strategy for the multiple container loading problem," *Eur. J. Oper. Res.*, vol. 223, no. 1, pp. 27–39, 2012.
[41] Y. He, Y. Wu, and R. de Souza, "A global search framework for practical three-dimensional packing with variable carton orientations," *Comput. Oper. Res.*, vol. 39, no. 10, pp. 2395–2414, 2012.
[42] R. Alvarez-Valdes, F. Parreño, and J. M. Tamarit, "A GRASP/Path relinking algorithm for two- and three-dimensional multiple bin-size bin packing problems," *Comput. Oper. Res.*, vol. 40, no. 12, pp. 3081–3090, 2013.
[43] L. Brunetta and P. Grégoire, "A general purpose algorithm for three-dimensional packing," *INFORMS J. Comput.*, vol. 17, no. 3, pp. 328–338, 2005.
[44] H. Wu, S. C. H. Leung, Y. W. Si, D. Zhang, and A. Lin, "Three-stage heuristic algorithm for three-dimensional irregular packing problem," *Appl. Math. Modelling*, vol. 41, pp. 431–444, Jan. 2016.
[45] R. Morabito and S. Morales, "A simple and effective recursive procedure for the manufacturer's pallet loading problem," *J. Oper. Res. Soc.*, vol. 49, no. 8, pp. 819–828, 1998.
[46] D. Zhang, Y. Peng, and S. C. H. Leung, "A heuristic block-loading algorithm based on multi-layer search for the container loading problem," *Comput. Oper. Res.*, vol. 39, no. 10, pp. 2267–2276, 2012.
[47] C. Paquay, M. Schyns, and S. Limbourg, "A mixed integer programming formulation for the three-dimensional bin packing problem deriving from an air cargo application," *Int. Trans. Oper. Res.*, vol. 23, nos. 1–2, pp. 187–213, 2016.
[48] M. Jahre and C. J. Hatteland, "Packages and physical distribution: Implications for integration and standardisation," *Int. J. Phys. Distrib. Logistics Manage.*, vol. 34, no. 2, pp. 123–139, 2004.
[49] M. Johnsson, "Packaging logistics—A value added approach," Ph.D. dissertation, Dept. Eng. Logistics, Lund Univ., Lund, Sweden, 1998.
[50] Z. L. Chen, "Integrated production and outbound distribution scheduling: Review and extensions," *Oper. Res.*, vol. 58, no. 1, pp. 130–148, 2010.
[51] F. T. S. Chan, H. K. Chan, and K. L. Choy, "A systematic approach to manufacturing packaging logistics," *Int. J. Adv. Manuf. Technol.*, vol. 29, nos. 9–10, pp. 1088–1101, 2006.
[52] D. Hellström and M. Saghir, "Packaging and logistics interactions in retail supply chains," *Packag. Technol. Sci.*, vol. 20, no. 3, pp. 197–216, 2007.
[53] T. H. S. Li, C.-J. Lin, P.-H. Kuo, and Y.-H. Wang, "Grasping posture control design for a home service robot using an ABC-based adaptive PSO algorithm," *Int. J. Adv. Robot. Syst.*, vol. 13, no. 3, p. 118, 2016, doi: 10.5772/64044.2016.
[54] *Amazon Japan Home Page*, accessed on Jan. 15, 2017. [Online]. Available: https://www.amazon.co.jp/
[55] *DHL Service Point Size and Price Guide*, accessed on Jan. 15, 2017. [Online]. Available: http://parcel.dhl.co.uk/dhl-service-point/size-and-price-guide
[56] *PCHome Online Home Page*, accessed on Jan. 20, 2017. [Online]. Available: http://24h.pchome.com.tw/
[57] V. Markovski, F. Xue, and L. Trajković, "Simulation and analysis of packet loss in user datagram protocol transfers," *J. Supercomput.*, vol. 20, no. 2, pp. 175–196, 2001.
[58] H. Lee, C. Y. Liu, C.-J. Lin, C.-F. Huang, R.-W. Deng, and T.-H. S. Li, "Implementation of real-time object recognition system for home-service robot by integrating SURF and BRISK," in *Proc. IEEE Int. Conf. Syst. Sci. Eng.*, Jul. 2014, pp. 273–278.

TZUU-HSENG S. LI (S'85–M'90) received the B.S. degree from the Tatung Institute of Technology, Taipei, Taiwan, in 1981, and the M.S. and Ph.D. degrees from National Cheng Kung University (NCKU), Tainan, Taiwan, in 1985 and 1989, respectively, all in electrical engineering.

Since 1985, he has been with the Department of Electrical Engineering, NCKU, where he is currently a Distinguished Professor. From 1996 to 2009, he was a Researcher with the Engineering and Technology Promotion Center, National Science Council, Tainan. From 1999 to 2002, he was the Director of the Electrical Laboratories with NCKU. From 2009 to 2012, he was the Dean of the College of Electrical Engineering and Computer Science with National United University, Miaoli City, Taiwan. Since 2009, he has been the Vice President of the Federation of International Robot-Soccer Association. Since 2014, he has been the Director of the Center for Intelligent Robotics and Automation with NCKU. His current research interests include artificial and biological intelligence and applications, fuzzy system and control, home service robots, humanoid robots, mechatronics, 4WIS4WID vehicles, and singular perturbation methodology.

Dr. Li received the Outstanding Automatic Control Award in 2006 from the Chinese Automatic Control Society (CACS) in Taiwan. He was a Technical Editor of the IEEE Transactions on Mechatronics and *ASME Transactions on Mechatronics*, and an Associate Editor of the *Asia Journal of Control*. He is currently an Associate Editor of the *International Journal of Electrical Engineering*, the *International Journal of Fuzzy Systems*, and the IEEE Transactions on Cybernetics. He was elected as the President of the CACS from 2008 to 2011 and the Robotics Society of Taiwan from 2012 to 2015. In 2008, he was elevated to CACS Fellow.

CHIH-YIN LIU received the B.S. degree from the Department of Social Work, National Taiwan University, Taipei, Taiwan, in 2003, and the M.S. degree from the Department of Psychology, National Chung Cheng University, Chia-Yi, Taiwan, in 2005. She is currently pursuing the Ph.D. degree with the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan. Her major research interests include intelligent control system, robot learning, home service robot, robot cooperation, and human–robot interaction.

PING-HUAN KUO received the B.S., M.S., and Ph.D. degrees from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2008, 2010, and 2015, respectively. His major research interests include fuzzy control, intelligent algorithms, humanoid robot, image processing, and robotic application.

NIEN-CHU FANG received the B.S. degree from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2015, where he is currently pursuing the M.S. degree. His major research interests include fuzzy control, intelligent system, humanoid robot, image processing, and robotic application.

CHENG-HUI LI received the B.S. degree from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2015, where he is currently pursuing the M.S. degree. His major research interests include home service robot, intelligent control system, robot vision, and object recognition.

CHING-WEN CHENG received the B.S. degree from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2015, where she is currently pursuing the M.S. degree. Her major research interests include home service robot, mechanical drawing, robotic manipulator motion control, and robot vision.

CHENG-YING HSIEH received the B.S. degree from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2016, where he is currently pursuing the M.S. degree. His major research interests include home service robot, intelligent control system, robot system and hardware design, robotic manipulator motion control, and mechanical drawing.

LI-FAN WU received the B.S. degree from the Department of Mechanical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2015, where he is currently pursuing the M.S. degree with the Department of Electrical Engineering. His major research interests include fuzzy control, intelligent system, humanoid robot, image processing, and robotic application.

JIE-JHONG LIANG received the B.S. degree from the Department of Mechanical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2016, where he is currently pursuing the M.S. degree with the Department of Electrical Engineering. His major research interests include home service robot, robotic manipulator motion control, and system modeling.

CHIH-YEN CHEN received the B.S. degree from the Department of Mechanical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2016, where he is currently pursuing the M.S. degree with the Department of Electrical Engineering. His major research interests include home service robot, robotic manipulator motion control, and system modeling.