Optimizing cooperative pallet loading robots: A mixed integer approach

Mattia Laurini[1], Luca Consolini[1], and Marco Locatelli[1]

***Abstract*—We consider a system with a metering conveyor that feeds a palletizing machine. Multiple robots cooperatively move packages to obtain a desired final pallet layout. We address the problem of finding the optimal sequence of manipulations and packages positions on the conveyor that allow to maximize production. Our main contribution is the reformulation of this operation as a mixed integer linear programming (MILP) problem, that we solve with a metaheuristic that exploits a standard commercial solver. We also present some numerical experiments on randomly generated problems.**

***Index Terms*—Planning, Scheduling and Coordination; Cooperating Robots; Task and Motion Planning**

I. Introduction

ROBOTIC palletizing systems have become widely present in manufacturing industrial facilities. In particular, in food, beverage and dairy industries, these machines allow for a faster production chain, ensuring an efficient palletizing process which results in a quicker storage and delivery of goods. The main task that these machines are designed to carry out is to form layers of products according to specific patterns and stack them on pallets.

In this paper, we propose a mixed integer linear programming model for capturing the essential features of a palletizing strategy. We will consider a system with a metering conveyor that feeds a palletizing machine with packages that are assumed to have all the same size. Multiple robots move packages in a cooperative way in order to obtain a desired final layout, which is given a priori, with the aim of maximizing production. We will formalize this problem with a model that is inspired by the traveling salesman problem (TSP) and, in particular, on its variations that contemplate the presence of multiple traveling salesmen (mTSP) and time windows (TSPTW), putting an additional constraint on the moment at which a traveling salesman is allowed to visit cities (see, e.g., [1], [2], [3], [4], [5]). Hence, the problem of choosing a palletizing strategy will be reformulated as a multiple traveling salesmen problem with time windows (mTSPTW), with additional constraints needed for avoiding collisions among packages during manipulations. In this setting, traveling salesmen represent the manipulators of the palletizing system, whilst cities represent positions on the conveyor belt. We call this problem the Pallet Pattern Placement Sequence Problem (3PSP).

Manuscript received: December, 18, 2020; Revised February, 2, 2021; Accepted April, 13, 2021.

This paper was recommended for publication by Editor Jingang Yi upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Regione Emilia-Romagna in project "ROBOT-A" of framework "POR FSE 2014/2020 Obiettivo Tematico 10", by OCME S.R.L. and by the Program "FIL 2019 – Quota Incentivante" of Università degli Studi di Parma and co-sponsored by Fondazione Cariparma.

[1]All authors are with the Department of Engineering and Architecture of the University of Parma, Parco Area delle Scienze, 181/A, 43124 Parma, Italy. {mattia.laurini, luca.consolini, marco.locatelli}@unipr.it.

Digital Object Identifier (DOI): see top of this page.

Literature review

In literature, a relevant amount of work has been done on bin packing (see, e.g., [6], [7], [8], [9], [10], [11], [12], [13]) and on optimizing space usage on a pallet, maximizing the amount of stacked products, which may be uniform in size or non-homogeneous (see, e.g., [11]), both in two dimensions (see, e.g, [6], [7], [8]) or three (see, e.g., [9], [10], [11], [12]). Many works are also devoted to the so called Pallet Loading Problem (see, e.g., [14], [15], [16], [17], [18]). However, in this work we do not focus on optimizing pallet usage as we assume a pallet layout is given a priori. Note that this situation holds true in many real life applications in which customers provide their own pallet layouts to palletizing machines producers. Here, we focus on developing a palletizing strategy for achieving a given pallet layout in order to maximize production. Some works introduce the 3PSP, however they do not mathematically model the problem and, hence, do not provide any specific algorithm for solving it (see, e.g., [19], [20], [21]). To the best of our knowledge, there are no works that explicitly address the problem of modeling the 3PSP in high-speed palletizing machines and provide an explicit solution algorithm.

Statement of contribution

The main contribution of this work is the reformulation of optimal planning for cooperative palletizing robots as a mixed integer linear programming (MILP) problem. To the best of our knowledge, the 3PSP has not been addressed in literature. Hence, this reformulation provides insight in the structure of this relevant industrial problem and a rigorous mathematical formalization. Another advantage given by the MILP formulation is the multitude of available solvers that can be used for computing a solution. In this work, we solved the resulting model with a solver for MILP problems (in particular, we used Gurobi [22]). Due to the complexity of the model and the large number of decision variables, we also developed a metaheuristic which allows us to solve problem instances with tens of packages (which is enough for addressing most industrial scenarios). In order to improve performances, in future works, we will use approaches specifically tailored to the problem at hand (for instance, dynamic programming), leveraging the rich literature on TSP and its variations.

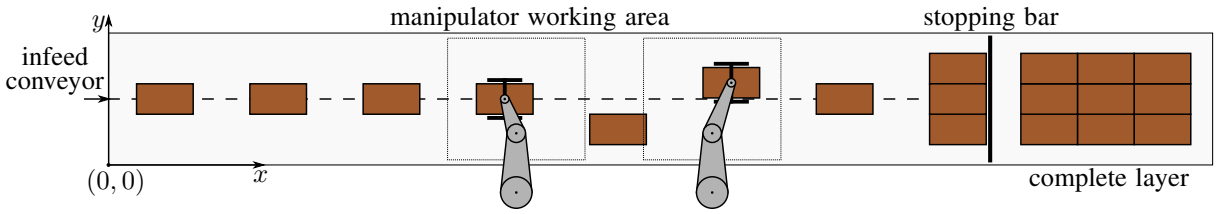


Fig. 1: A palletizing system with two manipulators, a stopping bar and a single-line infeed conveyor.

II. General description and assumptions

We consider the palletizing machine together with an inertial coordinate frame fixed on the moving belt (see Figure 1). In particular, assuming that the conveyor belt moves from left to right, the origin is placed at the position that the bottom-left corner of the belt occupies at the initial time. Moreover, we assume that the x-axis is parallel to the belt velocity and that the belt moves with constant speed w.

Products are transported by a conveyor belt and moved simultaneously by multiple robotic manipulators. Initially, packages are placed on the belt in a single track from an infeed conveyor. Multiple robotic manipulators move packages so that all of them reach the y-coordinate and orientation that correspond to an assigned pallet layout. Then, a stopping bar at the end of the conveyor belt and orthogonal to it aligns packages along x-axis before they move on the pallet. Note that, in general, there may be more than one stopping bar at the end of the conveyor belt and the outfeed conveyor; in this framework we assume there is only one stopping bar. The speed of the conveyor belt is a tuning parameter and is assumed to be constant during the palletizing process. Moreover, in high-speed palletizing machines, for speed of execution, packages are not lifted from the belt during manipulations. Hence, we must make sure that packages do not collide with each other.

Since the palletizing process can be the bottleneck of the whole packaging line, it is important to reduce the time needed to obtain the requested layout in order to maximize fully-loaded pallets production over time. Hence, in this paper, we aim at finding the package positions on the belt, the sequence of manipulations and the belt speed that allow to obtain the final layout in minimum time. In particular, we will show that this planning task can be represented as a linear optimization problem with binary variables.

To simplify the mathematical formulation of this problem, we make a number of assumptions:

i. Packages are identical to each other and rectangular.
ii. Manipulators working areas do not overlap.
iii. Each package is moved once.
iv. The space required by the end-effectors of manipulators is not considered.
v. At the end of the conveyor belt, there is a stopping bar orthogonal to it.
vi. With respect to the moving belt reference frame, packages are manipulated along y-axis.
vii. Packages are not lifted from the belt during manipulations.
viii. Conveyor belt speed w is a fixed parameter.

Without these assumptions, the resulting model would be very complex and it would be difficult to present it in a concise and readable way.

In particular, assumption i. can be relaxed in order to consider non-identical packages or groups of packages with different shapes. However, this would complicate the model in terms of number of constraints to be presented and make it more difficult to grasp the main features of this formulation. Nonetheless, the benefit from removing this assumption in regards of the model is twofold. Having non-identical packages implies that there need be precedence constraints between packages which depend on the final pallet layout. These constraints would restrict the feasible region of the model allowing for a faster search for a solution. Non-identical packages also means that we can consider groups with different shapes of identical packages which, from the model perspective, are considered nothing but packages with different size. Hence, grouping packages together, whenever possible and according to the manipulators mechanical limitations, reduces the size of the model.

Note that assumption ii. holds true for many palletizing machines and would not be restrictive in several industrial contexts. Relaxing this assumption would not complicate per se the collision avoidance constraints between packages that we will introduce later on. However, it would require the handling of collisions avoidance between manipulators whenever any of these is not performing a manipulation. We believe relaxing assumption ii. could be done within the mixed integer linear programming approach, however, we have not yet explored this generalization.

Regarding assumption iii., its relaxation, even though interesting, would require a non-trivial modelization effort which we think would result in a drastically more complicated model. Probably, tailored solution strategies would be necessary in order to solve such a model within a reasonable amount of time with respect to the application.

Assumption iv. may seem like a restrictive one, however, from the model perspective, considering the end-effectors size would just result in larger collision avoidance areas. The nature of the collision avoidance constraints would be the same; they would just involve additional parameters (known a priori, regarding the end-effectors size) which would just result in less readable constraints.

Assumption v. is a design requirement that holds true for many high-speed palletizing machines, which are usually equipped with one (or more) stopping bar(s) orthogonal to the belt velocity, placed at the end of the conveyor belt (and/or at the end of the outfeed conveyor, if present), that horizontally aligns packages and corrects misalignments along x-direction.

This implies that, in general, the x-coordinate of final package positions need not correspond to the x-coordinate of the final layout. In fact, x-coordinate mismatches are fixed through the activation of the stopping bar(s).

Assumption vi. states that for any manipulation of a package from an initial position (x_1, y_1) to a final one (x_2, y_2), it has to hold that $x_1 = x_2$; that is, we assume that, from the conveyor belt coordinate frame point of view, manipulators are only allowed to modify the y-coordinate of packages. Hence, all package manipulations are performed along paths that are straight segments parallel to the y-axis. Note that this assumption holds true in many industrial contexts and does not prevent the system from forming any given pallet layout due to the presence of stopping bar(s) at the end of the conveyor belt, as stated in assumption v.. Moreover, this assumption helps constraining final package positions to the initial ones removing a degree of freedom over final positions placement which can lead to a state space explosion of the model. This assumption could be lifted in future works introducing types of paths other than straight lines parallel to the y-axis. For instance, manipulations could occur along a straight segment which is not parallel to neither of the two axis, or it could be divided into two consecutive segments, one parallel to the x-axis and the other to the y-axis or viceversa. These last two types of paths could be particularly useful for avoiding collisions in case a straight line connecting the same pair of positions would cause a collision with other packages. However, the addition of these kind of paths would heavily impact the complexity of the problem and, therefore, we avoided considering them in this work.

Assumption vii. is another design requirement; in high-speed palletizing machines, for speed of execution, packages are not lifted from the belt during manipulations. This means that it is necessary to ensure collisions avoidance during manipulations.

Assumption viii. is a technical requirement, in fact, note that if we would consider w as an optimization variable, the resulting model would not be a MILP anymore as it would involve non-linear constraints. In particular, time window requirements involve w^{-1}. However, belt speed w can be optimized by bisection, running various instances of the optimization problem with different values of w, which is a fixed parameter for each optimization instance.

III. Preliminaries

We denote by N the number of packages and we consider a set of M manipulators, whose bases occupy a fixed position in the world's frame. With respect to the moving reference frame, at time t, the x-coordinate of the base of the m-th manipulator is $q_m - tw$, where q_m is the initial x-coordinate of the center of the m-th manipulator base with respect to the moving frame. Note that, with respect to the moving frame, the bases of the manipulators move along the x-axis with negative speed $-w$, in the opposite direction of the conveyor belt velocity.

We assume that the operating space of each manipulator is a rectangle parallel to the belt. On the x-axis, the rectangle is centered at the base of the manipulator and has size $2W$, with $W > 0$. On the y-axis, it covers the whole width of the belt (see Figure 1). With respect to the moving frame, at time t, the operating space of the m-th manipulator along the x-axis is the interval $[q_m - tw - W,\ q_m - tw + W]$.

Formulation as a modified multiple Traveling Salesmen Problem with Time Windows (mTSPTW)

Mathematically, we can represent the problem at hand as a mTSPTW with additional constraints.

As we mentioned in the Introduction, in this model, traveling salesmen represent manipulators, whilst cities represent positions on the conveyor belt; some positions are associated to packages initial positions on the conveyor belt, others are associated to packages final positions, according to a given pallet layout. In this way, traveling salesmen are manipulators whose task is to visit all initial package positions for picking packages up and placing them at final positions, in order to complete a pallet layout.

First of all, we define a graph whose nodes are associated to initial and final package positions and whose edges represent the allowed transitions. Namely, we define a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, in which the node set is $\mathcal{V} = \mathcal{O} \cup \mathcal{I} \cup \mathcal{F} \cup \mathcal{R}$. Set $\mathcal{O} = \{\mathcal{O}_1, \dots, \mathcal{O}_M\}$ represents the initial resting positions of the M manipulators, nodes in $\mathcal{I} = \{\mathcal{I}_1, \dots, \mathcal{I}_N\}$ are associated to the initial positions of the N packages on the belt, $\mathcal{F} = \{\mathcal{F}_1, \dots, \mathcal{F}_N\}$ refers to packages final positions and $\mathcal{R} = \{\mathcal{R}_1, \dots, \mathcal{R}_M\}$ represents manipulators final resting positions. Define, also, $\mathcal{N} = \mathcal{I} \cup \mathcal{F}$.

The edge set $\mathcal{E}$ is defined as follows:

- All nodes in $\mathcal{O}$ are connected to all nodes in $\mathcal{I}$, that is, manipulators can move from their initial resting position to an initial position on the belt occupied by a package.
- All nodes in $\mathcal{I}$ are connected to all nodes in $\mathcal{F}$. These edges represent the motions in which manipulators move a package from an initial to a final position.
- All nodes in $\mathcal{F}$ are connected to all nodes in $\mathcal{I}$.

These edges represent the motions with which, after having moved a package, manipulators return to an initial position to start another manipulation.

- All nodes in $\mathcal{F}$ are also connected to all nodes in $\mathcal{R}$. These edges represent those motions in which, after having moved a package, a manipulator goes to its final resting position.
- Each node in $\mathcal{O}$ is also connected to a corresponding node in $\mathcal{R}$. These edges represent the cases in which a manipulator does not move any package, so that it goes directly from its initial resting position to its final one.

Note that if a manipulator travels an edge in the last group (i.e., from $\mathcal{O}$ to $\mathcal{R}$), then it does not perform any manipulation, it is useless to the palletizing task and could be removed from the plant. Hence, edges of the last group should not be travelled in any appropriate solution.

In this way, the subgraph given by $(\mathcal{N}, \{e \in \mathcal{E} \mid e \in \mathcal{N} \times \mathcal{N}\})$ is a bipartite graph, since manipulators are only allowed to move from an initial package position to a final one, (i.e., they are only allowed to perform a manipulation), and from a final package position to an initial one, (i.e., after having performed a manipulation, they have to move to a new initial package position to perform a new manipulation).

At time $t = 0$, all manipulators occupy their initial position

in $\mathcal{O}$ and, at the end, each manipulator is required to be at its final resting position in $\mathcal{R}$. The sequence of movements of the m-th manipulator describes a path on the graph that starts at $\mathcal{O}_m$. Then, the manipulator has two options: it may go directly to the corresponding final resting position $\mathcal{R}_m$ (in this case, the manipulator does not move any package). Otherwise, the manipulator travels to a node in $\mathcal{I}$, then travels alternately between nodes in $\mathcal{F}$ and $\mathcal{I}$, and then, after having visited one last node in $\mathcal{F}$, goes to its final resting position $\mathcal{R}_m$.

As a clarifying example, consider a system with $N = 3$ packages and $M = 2$ manipulators. The associated graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with node set $\mathcal{V} = \{\mathcal{O}_1, \mathcal{O}_2, \mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{R}_1, \mathcal{R}_2\}$, is depicted in Figure 2. The manipulator at initial resting position $\mathcal{O}_1$ moves a package from initial position $\mathcal{I}_1$ to final position $\mathcal{F}_1$, then picks up another package at initial position $\mathcal{I}_2$ to drop it off at final position $\mathcal{F}_2$ and finally goes to its final resting position $\mathcal{R}_1$. The manipulator at initial resting position $\mathcal{O}_2$ moves a package from initial position $\mathcal{I}_3$ to final position $\mathcal{F}_3$ and then moves to its final resting position $\mathcal{R}_2$. The paths of the two manipulators are highlighted in Figure 2 in blue and green, respectively.

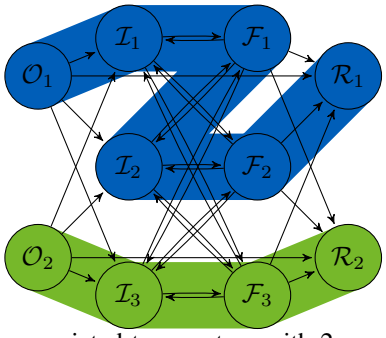


Fig. 2: Graph associated to a system with 2 manipulators and 3 packages with highlighted manipulators paths.

We associate an index to each element of $\mathcal{V}$. To this end, set $V = \{1, \dots, |\mathcal{V}|\}$ and define function $v : \mathcal{V} \to V$, such that $(\forall i \in \{1, \dots, M\})\ v(\mathcal{O}_i) = i$, $v(\mathcal{R}_i) = M + 2N + i$, and $(\forall i \in \{1, \dots, N\})\ v(\mathcal{I}_i) = M + i$ and $v(\mathcal{F}_i) = M + N + i$. Then, with respect to the node ordering given by v, graph $\mathcal{G}$ has the following adjacency matrix $A \in \{0,1\}^{|V| \times |V|}$

$$A = \begin{pmatrix} 0_{M,M} & 1_{M,N} & 0_{M,N} & I_{M,M} \\ 0_{N,M} & 0_{N,N} & 1_{N,N} & 0_{N,M} \\ 0_{N,M} & 1_{N,N} & 0_{N,N} & 1_{N,M} \\ 0_{M,M} & 0_{M,N} & 0_{M,N} & 0_{M,M} \end{pmatrix}, \tag{1}$$

where, for any $\ell, m \in \mathbb{N}$, $0_{\ell,m}$, $1_{\ell,m}$ and $I_{\ell,\ell}$ are the $\ell \times m$ zero matrix, the $\ell \times m$ matrix with all entries equal to 1 and the $\ell \times \ell$ identity matrix, respectively.

In order to keep track of the path followed by each manipulator, we introduce the following flow variables. For $i, j \in \mathcal{V}$ and $m \in \mathcal{M}$, define variable $X^m_{i,j} \in \{0,1\}$ as follows:

$$X^m_{i,j} = \begin{cases} 1, & \text{if manipulator } m \text{ moves from } i \text{ to } j, \\ 0, & \text{otherwise.} \end{cases}$$

Since each manipulator motion must correspond to an element of the edge set $\mathcal{E}$, for $i, j \in \mathcal{V}$, $m \in \mathcal{M}$, it must be $X^m_{i,j} \leq A_{v(i),v(j)}$.

For $i \in \mathcal{V}$, real variable T_i represents the time at which node i is visited by a manipulator. Given nodes $i, j \in \mathcal{V}$, we also associate to each edge in $\mathcal{E}$ from i to j a transition time variable t_{ij} which represents the time required to perform a transition from node i to node j. In the case of edges from $\mathcal{I}$ to $\mathcal{F}$, this also includes time τ, which is assumed to be a known constant, required to manipulators for picking up and releasing packages.

We define function $\rho : \mathcal{N} \to \mathbb{R}^2$ such that, for $i \in \mathcal{I}$, $\rho(i)$ represents the coordinates on the moving frame of the initial package position associated to node i. Similarly, for $j \in \mathcal{F}$, $\rho(j)$ represents the coordinates, with respect to the moving frame, of a final package position associated to node j.

At time t, the m-th manipulator can visit node $i \in \mathcal{V}$ only if $\rho(i)$ belongs to the m-th manipulator operating space. In other words, letting x be the x-coordinate of $\rho(i)$, it must hold that $x \in [q_m - tw - W,\ q_m - tw + W]$ or, equivalently, $tw \in [q_m - x - W,\ q_m - x + W]$. Setting $[a^m_i, b^m_i] = [w^{-1}(q_m - x - W),\ w^{-1}(q_m - x + W)]$, for all $i \in \mathcal{V}$ and $m \in \mathcal{M}$, if the m-th manipulator visits node i at time T_i, then it must hold that $T_i \in [a^m_i, b^m_i]$. This corresponds to a time window requirement on the node visit.

Recalling assumption vi., note that the path type is only relevant for the transitions from $\mathcal{I}$ to $\mathcal{F}$, since these are the only ones that can cause collisions. We assume that all other motions are done in the fastest possible way (in general, along a straight segment).

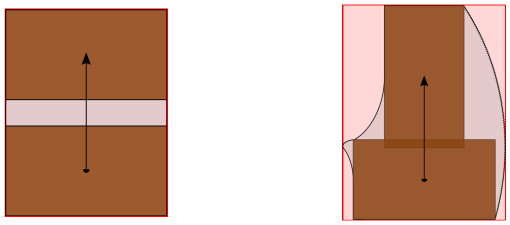
Collisions handling

As we mentioned earlier, packages are not lifted while being manipulated, so, in order to ensure collisions avoidance, we need to introduce the following quantities. For $i \in \mathcal{I}$, $j \in \mathcal{F}$, and $\ell \in \mathcal{N}$, define variable $c^\ell_{i,j} \in \{0,1\}$ such that

$$c^\ell_{i,j} = \begin{cases} 0, & \text{if a manipulation from } \rho(i) \text{ to } \rho(j) \text{ collides} \\ & \text{with a package placed at } \rho(\ell), \\ 1, & \text{otherwise.} \end{cases}$$

In other words, $c^\ell_{i,j} = 1$ if a package moved from the initial position associated to node i to the position in the final layout associated to node j does not collide with a package placed at the position corresponding to node ℓ. Otherwise, $c^\ell_{i,j} = 0$.

Note that variables $c^\ell_{i,j}$ depend on x-coordinates of the positions and initial-final positions association. Hence, they cannot be precomputed as problem parameters. However, there is a finite number of manipulation types which depend on the final pallet layout and on initial y-coordinates of the positions. In fact, packages get on the conveyor belt from a single track infeed conveyor and they are manipulated without modifying their x-coordinates. Hence, we can precompute the shapes associated to the portion of conveyor belt surface occupied during manipulations and, so, obtain the axis-aligned minimum bounding box (AABB) associated to each type of manipulation. The use of AABBs is particularly useful as it allows for modeling collision avoidance as a MIP problem (see, for instance, [23]). Moreover, a collision detection between a pair of AABBs can be decomposed along each axis and modeled with just one logical constraint per axis.



(a) Pure translation. (b) Rigid transformation.

Fig. 3: Manipulation types.

For instance, consider two AABBs B_1 and B_2 whose extremal points are $\{(x_{1_{\min}}, y_{1_{\min}}), (x_{1_{\max}}, y_{1_{\max}})\}$ and $\{(x_{2_{\min}}, y_{2_{\min}}), (x_{2_{\max}}, y_{2_{\max}})\}$, respectively, and assume B_1 is the AABB associated to a manipulation from node $i \in \mathcal{I}$ to $j \in \mathcal{F}$ and that B_2 is the AABB associated to a package at position $\ell \in \mathcal{N}$. Then, B_1 and B_2 do not intersect if and only if they are disjoint along x-axis or y-axis. This translates into the following constraint:

$$\begin{aligned} c^{\ell}_{i,j} = \quad & (x_{1_{\min}} \geq x_{2_{\max}}) \ \vee \ (x_{1_{\max}} \leq x_{2_{\min}}) \\ \vee \ & (y_{1_{\min}} \geq y_{2_{\max}}) \ \vee \ (y_{1_{\max}} \leq y_{2_{\min}}). \end{aligned} \tag{2}$$

If $c^{\ell}_{i,j} = 1$, then $B_1 \cap B_2 = \varnothing$, otherwise $B_1 \cap B_2 \neq \varnothing$. Again, observe that logical constraints (2) can be converted into linear ones with binary variables by means of standard techniques in operation research (see, e.g., [23], [24]).

Note that AABBs associated to packages exactly represent them since packages are themselves axis-aligned rectangles. The same can be said for manipulations of packages that are pure translations. In fact, since manipulations maintain packages x-coordinate fixed, the conveyor belt surface occupied during a pure translation corresponds to an axis-aligned rectangle. Such a manipulation is depicted in Figure 3a, in which the corresponding AABB is highlighted in red. Conversely, in case of rotations, AABBs associated to such manipulations overestimate, in general, the surface of the conveyor belt covered during these kind of manipulations. An example of this type of manipulation is depicted in Figure 3b, in which the corresponding AABB is highlighted in red, whilst the actual manipulation surface is highlighted in grey.

To handle collisions, during each manipulation we must know which of the initial and final positions are currently occupied by a package. To this end, we define a set of binary variables as follows. Set $\mathcal{K} = \{0, \dots, N\}$, then, for $k \in \mathcal{K}, i \in \mathcal{N}$, define $s^k_i \in \{0, 1\}$ such that

$$s^k_i = \begin{cases} 1, & \text{if node } i \text{ is occupied after the } k\text{-th manipulation,} \\ 0, & \text{otherwise.} \end{cases}$$

Here, a *manipulation* corresponds to the transition of one manipulator from a node in $\mathcal{I}$ to a node in $\mathcal{F}$ and the k-th manipulation corresponds to the one that occupies the k-th position in temporal order. Observe that when $k = 0$, s^0_i, with $i \in \mathcal{N}$, represents the state of package positions before the first manipulation is performed.

Transitions from $\mathcal{I}$ to $\mathcal{F}$ correspond to actual package movements on the belt and are susceptible to collisions, while all other transitions correspond to motions in which the end-effectors of manipulators are lifted from the belt and cannot cause collisions. Variables s^k_i contain information about positions occupied by each package at each time a manipulation is performed. Note that, if $i \in \mathcal{I}$, $s^0_i = 1$ and $s^N_i = 0$, whilst, if $j \in \mathcal{F}$, $s^0_j = 0$ and $s^N_j = 1$, since, initially, all nodes in $\mathcal{I}$ are occupied and all nodes in $\mathcal{F}$ are empty, and after all N manipulations the situation is reversed, with all nodes in $\mathcal{I}$ being empty and all nodes in $\mathcal{F}$ being occupied.

For $i \in \mathcal{I}$, $k \in \mathcal{K} \setminus \{0\}$, define binary variable

$$p^k_i = s^{k-1}_i - s^k_i. \tag{3}$$

Then, $p^k_i = 1$ if and only if the k-th manipulation moves the package at the initial position associated to node i. Conversely, for $j \in \mathcal{F}$, $k \in \mathcal{K} \setminus \{0\}$, define binary variable

$$p^k_j = s^k_j - s^{k-1}_j. \tag{4}$$

Then, $p^k_j = 1$ if and only if the k-th manipulation moves a package to the final position associated to node j.

IV. Problem formulation

Let us start presenting the model constraints associated to the core of an mTSP model.

$$\sum_{j \in \mathcal{N} \cup \mathcal{R}} \sum_{m \in \mathcal{M}} X^m_{i,j} = 1, \qquad i \in \mathcal{N} \cup \mathcal{O}, \tag{5}$$

$$\sum_{i \in \mathcal{N} \cup \mathcal{O}} \sum_{m \in \mathcal{M}} X^m_{i,j} = 1, \qquad j \in \mathcal{N} \cup \mathcal{R}, \tag{6}$$

$$\sum_{i \in \mathcal{N} \cup \mathcal{O}} X^m_{i,j} - \sum_{i \in \mathcal{N} \cup \mathcal{R}} X^m_{j,i} = 0, \qquad j \in \mathcal{N}, m \in \mathcal{M}, \tag{7}$$

$$X^m_{i,j} \leq A_{v(i),v(j)}, \qquad i, j \in \mathcal{V}, m \in \mathcal{M}, \tag{8}$$

$$X^m_{i,j} \in \{0, 1\}, \qquad i, j \in \mathcal{V}, m \in \mathcal{M}. \tag{9}$$

Condition (5) ensures that every node $i \in \mathcal{N} \cup \mathcal{O}$ has one and only one outgoing arc, considering every possible destination node $j \in \mathcal{N} \cup \mathcal{R}$ and any manipulator $m \in \mathcal{M}$. Similarly, condition (6) ensures that every node $j \in \mathcal{N} \cup \mathcal{R}$ has one and only one ingoing arc. Condition (7) is a flow conservation constraint, namely, for any $j \in \mathcal{V}$, it ensures that every manipulator can leave node j if and only if it has just reached it. Condition (8) ensures that the edges travelled by manipulators belong to edge set $\mathcal{E}$, where A is the adjacency matrix defined in (1). Finally, condition (9) is a binary requirement.

Let us now introduce the constraints associated to the state of the system.

$$s^k_i \geq s^{k+1}_i, \qquad i \in \mathcal{I}, k \in \mathcal{K} \setminus \{N\}, \tag{10}$$

$$s^k_j \leq s^{k+1}_j, \qquad j \in \mathcal{F}, k \in \mathcal{K} \setminus \{N\}, \tag{11}$$

$$\sum_{k \in \mathcal{K} \setminus \{0\}} p^k_i p^k_j = \sum_{m \in \mathcal{M}} X^m_{i,j}, \qquad i \in \mathcal{I}, j \in \mathcal{F}, \tag{12}$$

$$s^k_i \in \{0, 1\}, \qquad i \in \mathcal{N}, k \in \mathcal{K}, \tag{13}$$

$$s^0_i = 1, \quad s^N_i = 0, \qquad i \in \mathcal{I}, \tag{14}$$

$$s^0_j = 0, \quad s^N_j = 1, \qquad j \in \mathcal{F}. \tag{15}$$

Constraint (10) states that, for each $i \in \mathcal{I}$, variable s^k_i is monotone non-increasing with respect to k, since all packages at initial positions are, eventually, moved. Similarly, condition (11) state that, for each $j \in \mathcal{F}$, variable s^k_j is monotone

non-decreasing, since all final positions are initially empty and, at the end, occupied. Constraint (12) ensures that a transition from node $i \in \mathcal{I}$ to node $j \in \mathcal{F}$ can occur if and only if exactly one manipulator transitions from i to j. Note that p_i^k and p_j^k in constraint (12) are defined as in (3) and (4), respectively. Also, note that the non-linearity in the left-hand-side of this condition can be removed by introducing a set of additional N^2 binary variables representing the value of products $p_i^k p_j^k$. Condition (13) is a binary requirement. Finally, conditions (14) and (15) are the already discussed initial and final conditions on state variables s.

Now, let us introduce those constraints which represent the TSPTW part of the model.

$$(T_i + t_{ij} - T_j) \sum_{m \in \mathcal{M}} X_{i,j}^m \leq 0, \ i,j \in \mathcal{V}, \tag{16}$$

$$\sum_{m \in \mathcal{M}} a_i^m \sum_{j \in \mathcal{V}} X_{i,j}^m \leq T_i \leq \sum_{m \in \mathcal{M}} b_i^m \sum_{j \in \mathcal{V}} X_{i,j}^m, \ i \in \mathcal{N} \cup \mathcal{O}, \tag{17}$$

$$\sum_{m \in \mathcal{M}} a_j^m \sum_{i \in \mathcal{V}} X_{i,j}^m \leq T_j \leq \sum_{m \in \mathcal{M}} b_j^m \sum_{i \in \mathcal{V}} X_{i,j}^m, \ j \in \mathcal{R}, \tag{18}$$

$$\sum_{i \in \mathcal{I}} T_i \left(p_i^k - p_i^{k+1}\right) \leq 0, \ k \in \mathcal{K} \setminus \{0, N\}. \tag{19}$$

Condition (16) states that, if manipulator m travels from node i to node j, this has to be done in a time greater than or equal to t_{ij}. Note that this non-linear constraint can be replaced by a linear one by using Big-M constraints (this is a standard technique in Operations Research; see, for instance, [24]). Conditions (17) and (18) represent the time window constraints, enforcing the fulfillment of the lower bounds and upper bounds of the nodes visit time, respectively. They state that a node visit time has to fulfill the time window constraint associated to the manipulator that actually visits that node. Finally, condition (19) states that the visit time of an initial position at the k-th manipulation has to be smaller than the one of an initial position at the following manipulation. Also this non-linear constraint can be replaced by a linear one by means of Big-M constraints.

We now introduce the collision avoidance constraints.

$$X_{i,j}^m \leq \begin{matrix} 3 + c_{i,j}^\ell - s_\ell^{k-1} \\ -p_i^k - p_j^k \end{matrix}, \quad \begin{matrix} i \in \mathcal{I}, \ j \in \mathcal{F}, m \in \mathcal{M}, \\ k \in \mathcal{K} \setminus \{0\}, \ell \in \mathcal{N}, \ell \neq i, \end{matrix} \tag{20}$$

$$c_{i,j}^\ell \in \{0,1\}, \quad i,j \in \mathcal{V}, \ell \in \mathcal{N}. \tag{21}$$

Condition (20) ensures that a transition from a node $i \in \mathcal{I}$ to a node $j \in \mathcal{F}$ is not allowed if a package at node i is manipulated to node j in order k (i.e., $p_i^k = 1$ and $p_j^k = 1$), and, right before the k-th manipulation, there is a package that occupies node ℓ (i.e., $s_\ell^{k-1} = 1$), other than node i, that collides with this manipulation (i.e., $c_{i,j}^\ell = 0$). Condition (21) is a binary requirement. Finally, we recall that variables c_{ij}^ℓ also depend on constraints (2) which can be reformulated as a MIP problem as shown in [23].

The following constraint represents assumption vi.:

$$(x_j - x_i) \sum_{m \in \mathcal{M}} X_{i,j}^m = 0, \qquad i \in \mathcal{I}, j \in \mathcal{F}. \tag{22}$$

Constraint (22) ensures that if a manipulation is performed between a node $i \in \mathcal{I}$ and a node $j \in \mathcal{F}$, then the x-coordinates associated to these two nodes have to be equal.

We now introduce the last set of constraints, which involve only continuous variables.

$$t_{ij} = \tau + \frac{|y_i - y_j|}{w}, \quad i \in \mathcal{I}, j \in \mathcal{F}, \tag{23}$$

$$t_{ij} \geq \frac{|x_i - x_j| + |y_i - y_j|}{w}, \quad \begin{matrix} i \in \mathcal{V} \setminus \mathcal{I}, j \in \mathcal{V}, \\ \text{or } i \in \mathcal{I}, \ j \in \mathcal{V} \setminus \mathcal{F}, \end{matrix} \tag{24}$$

$$a_i^m = \frac{q_m - x_i - W}{w}, \quad i \in \mathcal{V}, m \in \mathcal{M}, \tag{25}$$

$$b_i^m = \frac{q_m - x_i + W}{w}, \quad i \in \mathcal{V}, m \in \mathcal{M}, \tag{26}$$

$$(x_j - x_k)\Pi_{j,k} \leq 0, \quad j,k \in \mathcal{F}, j \neq k. \tag{27}$$

Constraints (23) and (24) define transition times. In particular, (23) represents the transition time constraint when a manipulation is performed; the x-coordinate of the two positions has to be equal by (22) and so the transition time only depends on the offset along y-axis of the two positions and on a fixed term τ which represents the time required to manipulators for picking up and releasing packages. In fact, all values on the right-hand side of these constraints are problem data, so that in this case t_{ij} is a fixed value. Whilst, constraint (24) represents transition times in all remaning transitions. Note that when a robot is not performing a manipulation, we assume that it is moving along a straight segment connecting its current starting position to its destination position. Moreover, in order to avoid the presence of a square root in the right-hand-side of the constraint, in (24) the actual transition time is overestimated by using the L_1 distance instead of the L_2 one (i.e., the Euclidean distance). The right-hand side of these constraints is a convex piecewise linear function. Thus, each one of these constraints can be replaced by a pair of linear ones. Constraints (25) and (26) represent the lower and upper limit of the time window of each node for each manipulator, respectively. Lastly, constraint (27) represents the precedences between final positions. Since we assume the conveyor belt moves from left to right, the pallet layout is filled from right to left. Hence, assumption vi. requires precedences between final positions for ensuring the feasibility of the given layout. $\Pi \in \{0,1\}^{N \times N}$ is a precedence matrix which is precomputed based on the given pallet layout known a priori. For any $j,k \in \mathcal{F}$ such that $j \neq k$, $\Pi_{j,k} = 1$ if final position j cannot precede final position k along x-axis on the conveyor belt (i.e., $x_j \leq x_k$); otherwise, if the mutual position of the two is not relevant in regards of the feasibility of the layout, then $\Pi_{j,k} = 0$.

After applying the previously mentioned standard modifications (addition of auxiliary variables and use of Big-M constraints), Problem (5)–(27) becomes a MILP and can be solved by standard solvers for this class of problems.

V. Solution strategy

As we showed in the previous section, Problem (5)–(27) is a MILP and so it could be solved using any available MILP solver. However, due to the complexity of the problem and the amount of decision variables, which rapidly increase with the number of packages and manipulators, directly solving Problem (5)–(27) by means of a MILP solver alone may

result in unacceptably long computational times due to the difficulty of the solver in finding a feasible solution. In fact, the computational times for solving instances of this problem rapidly explode if the number of packages and manipulators slightly increases. Even though for high-speed palletizing machines the 3PSP need not be solved in real time, it is still desirable to keep computational times reasonably low. For this reason, we designed a metaheuristic which consists in separating the problem into two subproblems, which can be solved exactly by using MILP solvers with good performances compared to solving the initial problem altogether.

Basically, we separate the problem of assigning manipulations to robots, which essentially represents the MIP part of the model, from the one of optimizing package positions, which roughly corresponds to the LP part of the model.

For the MIP subroutine, in Problem (5)–(27) we fix initial package positions x_i, with $i \in \mathcal{I}$, as an input parmeter and solve (5)–(27) with objective function:

$$\min \max_{m,k \in \mathcal{M}} \left| \sum_{i \in I} \sum_{j \in F} X_{i,j}^{m} - \sum_{i \in I} \sum_{j \in F} X_{i,j}^{k} \right| . \tag{28}$$

Objective function (28) represents the maximum difference between the number of manipulations performed by each robot. It is desirable that all the available robots roughly perform the same number of manipulations in order to reduce the risk of failures due to the task overload of a robot and to equalize robots wear.

Conversely, for the LP subroutine, in Problem (5)–(27) we fix binary flow variables $X_{i,j}^{m}$, for $i \in I$, $j \in \mathcal{F}$ and $m \in \mathcal{M}$. As we already said, the ultimate objective is to maximize production. Given a conveyor belt speed w, maximizing production means finding the minimum-length sequence of package positions that allows manipulators to complete the given pallet layout moving all packages to their final position. Hence, the objective function for this subproblem is the following one:

$$\min \max_{i,j \in \mathcal{I}} |x_i - x_j| . \tag{29}$$

Objective function (29) represents the length of package positions sequence along x-axis, which we wish to minimize. Note that both objective functions (28) and (29) are convex piecewise linear, however, using standard procedures, it is possible to remove these non-linearities by adding linear constraints to the problem so that (28) and (29) are replaced by linear functions.

In this way, starting from well spaced initial positions along x-axis, we can fix the positions and solve the MIP subproblem; once this is solved, we fix the obtained manipulator tasks assignment variables and solve the LP subproblem, that is, we optimize package positions. We iterate this procedure alternating the solution of the two subroutines feeding the solution of a subroutine to the other and viceversa until convergence, that is, until the package positions sequence cannot be further shortened. Note that solutions obtained through this metaheuristic, despite being suboptimal, are still of good quality and obtained with low computational times, as we will see from the numerical experiments presented in the next section.

VI. Numerical experiments and conclusions

In this section, we present some numerical experiments that aim at showing how this approach provides good quality results for solving the 3PSP in high-speed palletizing machines, taking into account some of their kinematic properties. In the following, we consider a two-manipulator palletizing machine with a conveyor belt moving at speed $w = 0.45$ $\mathrm{m\,s^{-1}}$ and with a stopping bar at the end of it. The speed of the manipulators is 1.5 $\mathrm{m\,s^{-1}}$.

Let us consider a pallet layout example for presenting the manipulation sequences. In this scenario, the task is to form a layout, given in Figure 4, of 6 packages of size 0.410×0.204 m, whose positions are labeled with $\mathcal{F}_1, \ldots, \mathcal{F}_6$. Each package in the layout is assigned to a specific y coordinate, whilst the x coordinates are irrelevant in this regard as discussed in assumption v.. As long as packages are rearranged on the conveyor belt complying with precedence matrix Π of (27), the distance between packages along x-axis is not relevant. Precedences between final positions in the layout require that $\mathcal{F}_1$ and $\mathcal{F}_2$ need be placed after all other packages on the right, which can be placed in either order among each other. By solving Problem (5)–(27) associated to the layout of Figure 4, we obtain the package and manipulations sequences shown in Figures 5 and 6, which represent the first and the second robot manipulation sequence, respectively. Manipulators origin positions are labeled with $\mathcal{O}_1$ and $\mathcal{O}_2$, packages initial positions with $\mathcal{I}_1, \ldots, \mathcal{I}_6$, packages final positions with $\mathcal{F}_1, \ldots, \mathcal{F}_6$, and manipulators final resting positions with $\mathcal{R}_1$ and $\mathcal{R}_2$. Note that $\mathcal{R}_1$ and $\mathcal{R}_2$ coincide. The path and the AABBs associated to the first and the second manipulator are highlighted in blue and green, respectively.

We used the metaheuristic presented in Section V whose solution time, on this example, was 4.1569 s. Here, we solved the two subroutines of the metaheuristic with Gurobi [22], a commercial solver for mixed integer linear programs. We also solved the problem altogether with Gurobi (whose solution time was 2972 s) in order to compare the quality of the metaheuristic solution and found that in this case the metaheuristic provided the optimal solution of the problem. The reason why we picked a commercial solver is that in this work we did not focus on special purpose techniques for solving the proposed model with better performances, but this is indeed a topic for future research.

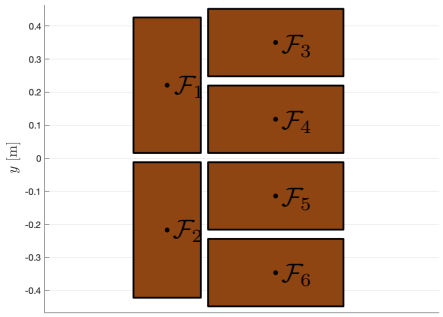


Fig. 4: Pallet layout.

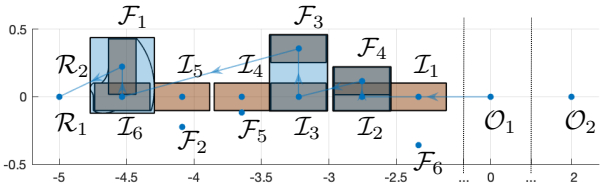


Fig. 5: First manipulator sequence.

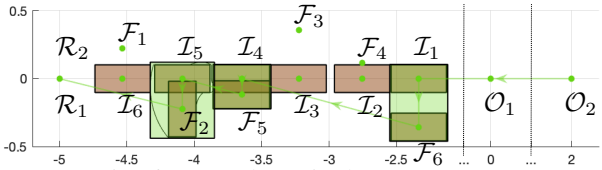


Fig. 6: Second manipulator sequence.

Finally, Figure 7 presents some computational time results for the proposed metaheuristic coupled with Gurobi for solving instances of Problem (5)–(27). For each increasing number of packages, we randomly generated 10 pallet layouts and solved the associated problems assuming there were two manipulators available. These layouts were generated by randomly placing non overlapping packages either parallel or perpendicular to the conveyor belt movement on a 2×2 m pallet. Figure 7 shows the box-and-whisker plot of the solution times for the previously described instances of Problem (5)–(27). It also shows the mean solution time for solving the same instances using Gurobi alone, for which we put a time limit of 3600 s (that also explains why the curve flattens between 7 and 9). Note that after 9 packages we stopped testing Gurobi alone as the benefits of the metaheuristic over Gurobi were clear. Note that we also performed tests on real life (and more regular) layouts provided by a company and, in that case, the solution times of the metaheuristic never exceeded 1000 s up to 21 packages.

We are aware of the combinatorial explosion this model may cause as packages number increases and we are developing techniques for addressing this issue. However, note that, for the specific application of this model, the actual number of packages generally does not go beyond a couple of tens. Moreover, the 3PSP is solved offline, so that computational times need not be compatible with real time applications, and computing times of the order of minutes are allowed. Finally, in order to improve performances, in future works, we will use approaches specifically tailored to the problem at hand (for instance, dynamic programming), leveraging the rich literature on the TSP problem and its variation.

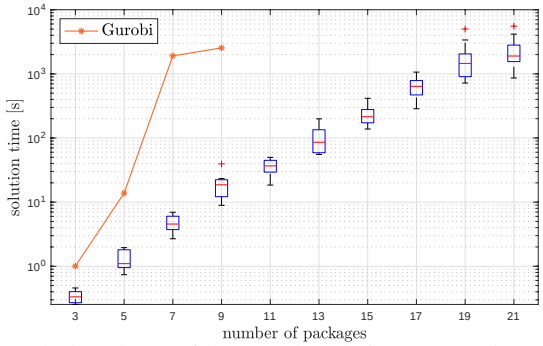


Fig. 7: Solution times of the metaheuristic compared to Gurobi for different numbers of packages.

ACKNOWLEDGMENT

The authors are grateful to the Associate Editor and the anonymous Reviewers for the suggestions which allowed to improve this work.

REFERENCES

[1] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations Research*, vol. 35, no. 2, pp. 254–265, 1987.

[2] M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, *Network Routing*, ser. Handbooks in Operation Research and Management Science. Elsevier, 1995, vol. 8.

[3] A. N. Letchford and R. W. Eglese, "The rural postman problem with deadline classes," *European Journal of Operational Research*, vol. 105, pp. 390–400, 1998.

[4] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, vol. 34, pp. 209–219, 2006.

[5] J. Nossack, B. Golden, E. Pesch, and R. Zhang, "The windy rural postman problem with a time-dependent zigzag option," *European Journal of Operational Research*, vol. 258, pp. 1131–1142, 2017.

[6] D. S. Johnson, "Fast algorithms for bin packing," *Journal of Computer and System Sciences*, vol. 8, no. 3, pp. 272–314, 1974.

[7] J. O. Berkey and P. Y. Wang, "Two-dimensional finite bin-packing algorithms," *Journal of the Operational Research Society*, vol. 38, no. 423–429, 1987.

[8] A. Soke and Z. Bingul, "Hybrid genetic algorithm and simulated annealing for two-dimensional non-guillotine rectangular packing problems," *Engineering Applications of Artificial Intelligence*, vol. 19, pp. 557–567, 2006.

[9] H. Dyckhoff, "A topology of cutting and packing problems," *European Journal of Operational Research*, vol. 44, pp. 145–159, 1990.

[10] G. Abdou and E. Lee, "Contribution to the development of robotic palletization of multiple box sizes," *Journal of Manufacturing Systems*, vol. 11, no. 3, pp. 160–166, 1992.

[11] C. S. Chen, S. M. Lee, and Q. S. Shen, "An analytical model for the container loading problem," *European Journal of Operational Research*, vol. 80, pp. 68–76, 1995.

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

[13] G. Fasano and J. D. Pintér, Eds., *Optimized Packings with Applications*. Springer, 2015, vol. 105.

[14] K. A. Dowsland, "An exact algorithm for the pallet loading problem," *European Journal of Operational Research*, vol. 31, pp. 78–84, 1987.

[15] ——, "Efficient automated pallet loading," *European Journal of Operational Research*, vol. 44, pp. 232–238, 1990.

[16] B. Ram, "The pallet loading problem: A survey," *International Journal of Production Economics*, vol. 28, pp. 217–225, 1992.

[17] E. Silva, J. F. Oliveira, and G. Wäscher, "The pallet loading problem: a review of solution methods and computational experiments," *Int. Transactions in Operational Research*, vol. 23, pp. 147–172, 2016.

[18] G. Scheithauer, *Introduction to Cutting and Packing Optimization Problems, Modeling Approaches, Solution Methods*, ser. Int. Series in Operations Research & Management Science. Springer, 2017.

[19] H. A. Khan and S. H. Masood, "Placement sequence methodology in pallet pattern formation in robotic palletisation," *Advanced Materials Research*, vol. 383–390, pp. 6347–6351, 2012.

[20] H. A. Khan, S. H. Masood, and A. Giecco, "An algorithm to determine placement sequence in robotic pallet pattern formation," *Advanced Materials Research*, vol. 403–408, pp. 3953–3958, 2012.

[21] S. H. Masood and H. A. Khan, "Development of pallet pattern placement strategies in robotic palletisation," *Assembly Automation*, vol. 34, no. 2, pp. 151–159, 2014.

[22] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2020.

[23] M. Fischetti and I. Luzzi, "Mixed-integer programming models for nesting problems," *Journal of Heuristics*, vol. 15, pp. 201–226, 2009.

[24] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, Inc., 1998.