Optimizing the packing strategy for parcel delivery vans

Louise Johanna Zwep

Cover image: Felix Reed (2020) [Edited by Louise Zwep (2023)]. *Courier delivering parcel.*
Retrieved from www.dreamstime.com.

Optimizing the packing strategy for parcel delivery vans

by

Louise Johanna Zwep

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday May 23, 2023 at 12:30 PM.

Student number:	4581733	
Project duration:	September 5, 2023 – May 23, 2023	
Thesis committee:	Prof. Dr. D. C. Gijswijt,	TU Delft, supervisor
	Dr. G. F. Nane,	TU Delft
	Ir. T. Jonker,	PostNL B.V.

Abstract

The increasing popularity of e-commerce has led to a greater emphasis on improving parcel delivery processes. Among the various stages of the delivery process, packing parcels into delivery vans affects the delivery time. The efficiency of delivery is optimized when each parcel is conveniently accessible upon arrival, thereby minimizing the requirement for additional repacking time. Therefore, streamlining parcel packing within delivery vehicles is a crucial aspect of improving overall delivery times in e-commerce. This thesis focuses on developing a packing solution that conforms to the Last-In-First-Out (LIFO) principle, which is defined as ensuring that a parcel can be accessed by the delivery driver as soon as they reach the destination of the parcel. This is described as the 3D-Bin Packing Problem with Loading Constraints (3L-BPP). To solve this problem, a formulation of a Mixed Integer Linear Program (MILP) has been developed. To improve the speed and accuracy of the solution, a novel placement heuristic has been created. This heuristic is derived from the established Distance to the Front-Top-Right Corner (DFTRC)-2 method and is designed to generate an initial solution.

Both the MILP and the heuristic are tested on a data set of 789 distinct rides, provided by a Dutch postal company. The results demonstrate that (a modified version of) the heuristic successfully generated an initial solution for all rides in the dataset, with 98.9% being found within 3 seconds, and for the remaining 1.1%, the inclusion of a Genetic Algorithm led to a solution being found within 90 seconds. By using the heuristic to establish an initial solution that is then refined through optimization techniques for the MILP, the findings indicate that this approach yields the best outcomes in minimizing the number of incorrectly positioned parcels.

Keywords: 3D-Bin Packing Problem, Mixed Integer Linear Program, Placement heuristics, LIFO

Preface

After almost seven years at the Technical University of Delft, this thesis marks the end of my academic journey. Before *delving* into the technical details, I would like to take a moment to appreciate the people around me during this time. A thank you to my family and to my friends from Contrast, Tarantula, Villa Verheven and Board 63 for making my time in Delft more than amazing. And of course to Sam for all his love and encouragement. A special thank you goes to my fellow 'strijders' from the study, who were able to get me through all the bad and good of mathematics, while making me laugh an endless times.

I would like to thank PostNL for giving me the opportunity to preform my thesis there, and the people at DBE for making me feel so welcome. I was treated as an equal team member from day one, which I really enjoyed. Taking part in the hackathon, training sessions, team days and 'superborrels' made doing my thesis much more than just studying. A very special thank you to my daily supervisor Tim, for all his time, knowledge and the fun moments together. I am super happy that you were (not so voluntarily) assigned to my project. I would also like to thank my supervisor from the TU Delft, Dion, for all the nice conversations about the research and for challenging me to keep on improving the model.

Louise Zwep
Delft, May 2023

Contents

List of Tables

List of Figures

Glossary

3D-BPP 3D-Bin Packing Problem

3D-SPP 3D-Strip Packing Problem

3L-BPP 3D-Bin Packing Problem with Loading Constraints

3L-CVRP Capacitated Vehicle Routing Problem with 3D Loading Constraints

ACLPPD Airline Container Loading Problem with Pickup and Delivery

CP Constraint Programming

CVRP Capacitated Vehicle Routing Problem

DBLF Deepest-Bottom-Left with Fill

DFTRC Distance to the Front-Top-Right Corner

ES Empty Spaces

GA Genetic Algorithm

LBBC Left-Bottom-Back Corner

LIFO Last-In-First-Out

LMD Last Mile Delivery

LP Linear Programming

MILP Mixed Integer Linear Program

1
Introduction

1.1. Problem description

From 2014 until 2021, the retail e-commerce sales worldwide grew from 1.3 to 5.2 billion US dollar, implying that the size quadrupled in only seven years. Expected is that in 2026 this number even grows to 8.2 billion US dollars [1]. The growth in retail e-commerce has a significant impact on the parcel delivery market, as a vast number of products are shipped through this channel. In the Netherlands, the number of parcels delivered went from 343 million in 2014 to 954 million in 2021 [2]. This increase calls for optimization in the process of parcel delivery in the e-commerce branch.

Parcels are more often than not going through a long process before reaching the consumer. Starting from collection at warehouses of the customer, multiple overlays at sorting centers, going from truck to truck, to finally end up on the doorstep of the consumer. The last part of this process, from the final sorting center to the delivery address is called the Last Mile Delivery (LMD). This part is crucial for any postal service, as this is the part where the company comes in contact with the consumer. One part of LMD is the packing process of the delivery van. In this thesis, this step is being investigated.

One of the unfavorable situations for parcel delivery services is that during the trip the delivery driver has to rearrange a part of the van to be able to reach a specific parcel. This situation is inconvenient both for the delivery driver, who has to handle additional tasks, and for the consumers, who have to wait longer to receive their parcel. Also, having to pack the van is stated as one of the most stressful obligations of the delivery drivers and especially hard for the new delivery drivers and those who just got assigned a new route [3].

In order to address these issues, it is important to devise a packing strategy for the parcel delivery van. For many parcel delivery services, the delivery sequence and parcel dimensions are known beforehand. This information can be used to create an optimized layout for the van through the use of exact and algorithmic mathematical methods. The research conducted in this thesis aims to provide a solution to the question:

Research question

What modifications can be made to existing parcel packing optimization models to a create feasible packing solution within delivery vans, given a specific set of parcels and delivery sequence?

In order to answer this question, sub-questions are formulated focusing on the different parts of the problem:

1. How can criteria be defined that make a packing feasible and can these criteria be expressed as a mathematical problem?

2. Is it possible to construct an exact model that solves the packing problem and to provide proof of its correctness?

3. What is the most effective heuristic solution method for finding a feasible packing, given the defined criteria?

4. How does the runtime performance and reliability of unloading ease compare between the exact model and the heuristic solution method?

This thesis is structured as follows: in the rest of this chapter a literature overview is set forth, in which an analysis is done on the previous work related to this problem. Then, in Chapter 2, the preliminaries needed for this research are given. In Chapter 3, the problem is described, and an analytical model is proposed to solve the problem, which addresses the first two sub-questions. The heuristic solution methods are discussed in Chapter 4, addressing the third sub-question. Chapter 5 outlines modifications made to both models and demonstrates how the heuristic approach serves as a foundation, while the analytical model is employed to optimize the outcome. The models are tested on an extensive dataset, and the results are examined in Chapter 6, answering the last question. Finally, Chapter 7 concludes the research, discusses its implications, and suggests options for future research.

1.2. Literature overview

As stated in the previous section, the optimization of container and van packing has garnered significant attention over the past few decades, owing to the increasing market of e-commerce since the advent of the internet and the rise in globalization and complexity of world markets [4]. Consequently, the relevance of container packing has increased in academic research. Although there is no specific model that addresses the problem of generating a feasible packing for a parcel delivery van, a substantial amount of research has been conducted on the 3D-Bin Packing Problem (3D-BPP), which serves as a strong foundation for solving the given problem. Therefore, this section covers the literature pertaining to the 3D-BPP and related problems.

The 3D-BPP is defined as follows: Given are a set of heterogeneous items that are rectangular in shape and an infinite set of identical bins or containers that are also rectangular. The objective of the problem is to pack all the items into the minimum number of bins. Two assumptions are made in the basis of the problem: the items should not be placed diagonally in the van, thus that they are packed with each edge parallel to one of the bin edges, and none of the items is (in any direction) larger than the containers [5].

The first known interest in this problem was in 1971 by Brown [6] in his book on optimum packing and depletion. After that, the problem was mentioned several times, but each time with a different name or description. In 1990 Dyckhoff [7] gave an overview of the research done so far and categorized all the variations of the problem that were given. A more structured approach to research in this area was then developed. This categorization will also be used in this thesis.

The 3D-bin packing problem is, naturally, a generalization of the 1D-bin packing problem. This problem is strongly NP-hard, making 3D-BPP also strongly NP-hard [5]. Many different methods have been used to solve 3D-BPP, going from analytical to heuristic solution methods.

Changes in the constraints or assumptions can be made to make 3D-BPP more realistic. Chen *et al.*[8] did this in 1995 by removing the assumption that the containers are homogeneous. This leads to the objective to minimize the unused space, for which they presented a Mixed Integer Linear Program (MILP). This model is described in Chapter 2 and is also discussed in the model description in Chapter 3. This model can be seen as the basis on which many subsequent articles have been built.

Another connection with 3D-BPP was made by Kantorovich [9] in 1939, associating it with cutting problems. Later Dyckhoff [7] and Faina [10] established the same parallels. The important relationship between the two, as Faina noted, comes from the duality of the problems: packing boxes into a container can be seen as cutting the space of the container into pieces [10].

This gave rise to the 3D-Strip Packing Problem (3D-SPP). In this variation of 3D-BPP the height of the container (the strip) is used as a variable instead of a parameter. The objective is to find a load of items that makes the height as small as possible. This approach to the problem was used by George and Robbinson

[11], Maarouf [12] and Bischoff [13] in heuristic solution methods for 3D-BPP, where they were able to find an optimal position for 500 boxes in a few minutes. They did so by creating slices of minimal height in the container and then pushing them together.

A considerable proportion of articles on bin packing modify the problem to minimise the Capacitated Vehicle Routing Problem (CVRP). Here they only include the packing procedure as a constraint [4] [14] [15]. Genreau *et al.* [16] were the first to introduce the Capacitated Vehicle Routing Problem with 3D Loading Constraints. This was done by creating a 3D-packing space and assigning items to coordinates within this space. They added four constraints related to the packing strategy:

1. Fixed vertical orientation: The height of the items is displayed on the z-axis. Items can still rotate 90° in the xy-plane.

2. Fragility: If an item is marked as fragile, only other fragile items can be placed on top of it.

3. Sufficient support area: The support area of an item is defined as the combined area of its bottom surface and the top surface of any overlapping items at the same height on the xy-plane. This area should be at least a given percentage of the item size.

4. Last-In-First-Out (LIFO) policy: If a customer is visited earlier than the others, the item(s) belonging to that customer should be accessible from the back of the van. This means that it should be placed above the other items (so higher on the z-axis) or in front of them (so further away from the origin on the y-axis).

These constraints are all considered for this research and adopted when relevant to the model. More about the constraints used can be found in Section 3.1.

A rather different approach to the problem was taken by Lurkin in 2015 [17], who introduced the Airline Container Loading Problem with Pickup and Delivery (ACLPPD). In this problem, the input is a set of containers, a cargo aircraft and a flight plan. An important difference in the approach is that this aircraft is compartmentalized. Lurkin created a MILP in which each container is assigned to one of the compartments in the aircraft. Because it is an assignment problem and does not work with x-, y- and z-coordinates, it is significantly more efficient than the model created by Chen *et al.*. The paper presents an instance of 128 containers solved within 10 minutes. This outcome is noteworthy considering the added complexity of ensuring that the containers are accessible from the back in a specific order during placement.

The most common heuristic method for solving 3D-BPP is the Deepest-Bottom-Left with Fill (DBLF) algorithm. The first step in formulating this algorithm was taken by Jakobs in 1996 [18], when he devised an algorithm for packing a 2D bin. In the Bottom-Left algorithm, each item is placed as much as possible towards the bottom and then as much as possible to the left. The elegance of this algorithm is that the running time is only $\mathcal{O}(n^2)$, where n is the number of items to pack. The disadvantage of this algorithm is that it leaves a lot of empty space between the packed items.

An extension was made by Hopper [19], who made the Bottom-Left with Fill algorithm. In this algorithm, the item fills the smallest possible area of the bin. This unfortunately makes the algorithm $\mathcal{O}(n^3)$, but improves the results of creating a compact packing. Karabulut and İnceoğlu [20] then converted this to the 3D variant. Here they listed the empty volumes by size. Then, for each item, they place it in the smallest possible empty volume and update the volume sizes. This is a very basic way of filling the box, but together with a hybrid genetic algorithm to find the input order, it gives some favorable results. The pseudo-code of this algorithm is given in Section 2.4.1.

This thesis presents a novel approach to find a feasible packing for parcel delivery vans, referred to as the 3D-Bin Packing Problem with Loading Constraints (3L-BPP). This problem combines elements of both the 3D-Bin Packing Problem (3D-BPP) and the Capacitated Vehicle Routing Problem with 3D Loading Constraints (3L-CVRP). To address this problem, an analytical model is introduced that provides an exact solution. Additionally, several heuristic methods are explored, and the best one is selected, and also changed to be more fitting for the problem. Finally, the analytical model is combined with the created heuristic method to produce a solution that is as optimal as possible. The proposed models are tested and verified to ensure their accuracy and effectiveness in solving the 3L-BPP.

2 Preliminaries

2.1. Mixed Integer Linear Program (MILP)

To tackle the problem outlined in Section 1.1, the powerful technique of Linear Programming (LP) is employed. This approach is rooted in the Operational Research subfield of mathematics, and involves formulating a problem in mathematical terms and then seeking an optimal or feasible solution. A brief introduction on this topic is given.

2.1.1. General model

According to Vanderbei's book [21], the concept of LP can be described as follows: In this method, the values of certain variables **x** need to be determined. The objective is to maximize the objective function, which is a linear function of the decision variables. Additionally, there are constraints that the variables must satisfy. These constraints are bounds on the linear combinations of the variables. Therefore, the standard form of an LP is as follows, with **c** and **b** a vector and A a matrix:

$$\begin{aligned} \text{Maximize} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{Such that} \quad & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

The variables in an LP are continuous. An Integer Linear Program (ILP) is a program of the same form, but where all variables **x** are integer. Next to that, there is the Mixed Integer Linear Program (MILP), where at least one variable is non-integer. Several techniques and specialized software packages are available to optimize these types of problems.

2.2. MILP for the 3D-Bin Packing Problem

The 3D-Bin Packing Problem (3D-BPP) concerns the problem of packing a given set of items into a set of bins. A description of this poblem can be found in Section 1.2. For the remainder of this thesis, the terms "vans" and "parcels" refer to the bins and items, respectively. A MILP for this problem was given by Chen *et al.* in 1995 [8]. This model is based on describing the vans as a 3D-space and assigning each parcel an (x, y, z)-coordinate within this space. In this section this model is presented. The parameters and variables are given first, followed by the objective function and constraints.

2.2.1. Parameters

Table 2.1 displays the parameters of the MILP for the 3D-BPP. These parameters must be defined in advance. Specifically, the length of a parcel refers to its longest dimension, the height to its shortest dimension, and the width to the remaining dimension. So, this denotes that $p_i \geq q_i \geq r_i$.

2.2.2. Variables

In Table 2.2, the variables of the model are presented. The value of these variables will be determined by the model itself.

Table 2.1: Parameters used in the Mixed Integer Linear Program of the 3D-Bin Packing Problem

Symbol	Description
P	The set of parcels to be packed
N	Total number of parcels to be packed in the van, equal to $\|P\|$
V	Set of vans that are available
m	Total number of vans that are available, equal to $\|V\|$
M	A large number for modeling purposes
(p_i, q_i, r_i)	Parameters describing the length, width and height of parcel i
(L_j, W_j, H_j)	Parameters describing the length, width and height of van j

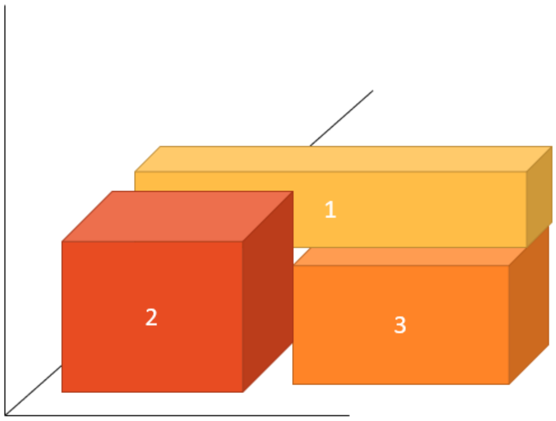


Figure 2.1: Parcels 1, 2 and 3 in a 3-dimensional field. Here the values for the variables for relative position are $a_{23} = d_{12} = f_{13} = 1$ and 0 for the rest. This means that parcel 2 is on the left of parcel 3, parcel 1 is in front of parcel 2 and parcel 1 is on top of parcel 3. The corner where the axes come together is the origin.

The term “parcel i to the left of k" denotes that parcel i is located closer to the origin on the x-axis than parcel k. Conversely, “parcel i on the right side of k" means that parcel i is further from the origin on the x-axis than parcel k. When “parcel i is in front of parcel k”, it indicates that parcel i is located further away from the origin than parcel k on the y-axis, which may seem counter-intuitive at first. Lastly, if “parcel i is below parcel k”, it means that parcel i is located closer to the origin on the z-axis than parcel k. The variables $a_{ik},..., f_{ik}$ denote the relative position variables.

An illustration of three parcels and their relative positions is shown in Figure 2.1. In this example, the variables a_{23}, d_{12}, and f_{13} are equal to 1, while all other variables that determine relative position are 0. It is important to note that variables for relative position are only defined for $i < k$. The Left-Bottom-Back Corner (LBBC) of a parcel is represented by the (x, y, z)-coordinates of the corner closest to the origin.

Table 2.2: Variables used in the Mixed Integer Linear Program of the 3D-Bin Packing Problem

Symbol	Description
(x_i, y_i, z_i)	Continuous variables indicating the (x,y,z)-coordinates of the Left-Bottom-Back Corner (LBBC) of parcel i
(l_{xi}, l_{yi}, l_{zi})	Binary variables indicating whether the length of parcel i is parallel to either the x-, y- or z-axis.
(w_{xi}, w_{yi}, w_{zi})	Binary variables indicating whether the width of parcel i is parallel to either the x-, y- or z-axis
(h_{xi}, h_{yi}, h_{zi})	Binary variables indicating whether the height of parcel i is parallel to either the x-, y- or z-axis
s_{ij}	1 if parcel i is assigned to van j 0 otherwise
n_j	1 if van j is used 0 otherwise
a_{ik}	1 if parcel i is on the left side of parcel k 0 otherwise
b_{ik}	1 if parcel i is on the right side of parcel k 0 otherwise
c_{ik}	1 if parcel i is behind parcel k 0 otherwise
d_{ik}	1 if parcel i is in front of parcel k 0 otherwise
e_{ik}	1 if parcel i is below parcel k 0 otherwise
f_{ik}	1 if parcel i is above parcel k 0 otherwise

2.2.3. Objective and Constraints

Underneath, the objective function and the constraints for the MILP of the 3D-BPP are presented:

$$\text{Minimize} \sum_{j=1}^{m} L_j \cdot W_j \cdot H_j \cdot n_j - \sum_{i=1}^{N} p_i \cdot q_i \cdot r_i \tag{2.1}$$

$$\text{Subject to } x_i + p_i \cdot l_{xi} + q_i \cdot w_{xi} + r_i \cdot h_{xi} \leq x_k + (1 - a_{ik}) \cdot M \qquad \forall i, k \in P : i < k, \tag{2.2}$$

$$x_k + p_k \cdot l_{xk} + q_k \cdot w_{xk} + r_k \cdot h_{xk} \leq x_i + (1 - b_{ik}) \cdot M \qquad \forall i, k \in P : i < k, \tag{2.3}$$

$$y_i + q_i \cdot w_{yi} + p_i \cdot l_{yi} + r_i \cdot h_{yi} \leq y_k + (1 - c_{ik}) \cdot M \qquad \forall i, k \in P : i < k, \tag{2.4}$$

$$y_k + q_k \cdot w_{yk} + p_k \cdot l_{yk} + r_k \cdot h_{yk} \leq y_i + (1 - d_{ik}) \cdot M \qquad \forall i, k \in P : i < k, \tag{2.5}$$

$$z_i + r_i \cdot h_{zi} + q_i \cdot w_{zi} + p_i \cdot l_{zi} \leq z_k + (1 - e_{ik}) \cdot M \qquad \forall i, k \in P : i < k, \tag{2.6}$$

$$z_k + r_k \cdot h_{zk} + q_k \cdot w_{zk} + p_k \cdot l_{zk} \leq z_i + \big(1 - f_{ik}\big) \cdot M \qquad \forall i, k \in P : i < k, \tag{2.7}$$

$$a_{ik} + b_{ik} + c_{ik} + d_{ik} + e_{ik} + f_{ik} \geq s_{ij} + s_{kj} - 1 \qquad \forall i, k \in P : i < k, \tag{2.8}$$

$$\sum_{j=1}^{m} s_{ij} = 1 \qquad \forall i \in P \tag{2.9}$$

$$\sum_{i=1}^{N} s_{ij} \leq M \cdot n_j \qquad \forall j \in V, \tag{2.10}$$

$$x_i + p_i \cdot l_{xi} + q_i \cdot w_{xi} + r_i \cdot h_{xi} \leq L_j + (1 - s_{ij}) \cdot M \qquad \forall i \in P, \tag{2.11}$$

$$y_i + q_i \cdot w_{yi} + p_i \cdot l_{yi} + r_i \cdot h_{yi} \leq W_j + (1 - s_{ij}) \cdot M \qquad \forall i \in P, \tag{2.12}$$

$$z_i + r_i \cdot h_{zi} + q_i \cdot w_{zi} + p_i \cdot l_{zi} \leq H_j + (1 - s_{ij}) \cdot M \qquad \forall i \in P \tag{2.13}$$

$$l_{xi} + l_{yi} + l_{zi} = 1 \qquad \forall i, j \in P, \tag{2.14}$$

$$w_{xi} + w_{yi} + w_{zi} = 1 \qquad \forall i, j \in P, \tag{2.15}$$

$$h_{xi} + h_{yi} + h_{zi} = 1 \qquad \forall i, j \in P, \tag{2.16}$$

$$l_{xi} + w_{xi} + h_{xi} = 1 \qquad \forall i, j \in P, \tag{2.17}$$

$$l_{yi} + w_{yi} + h_{yi} = 1 \qquad \forall i, j \in P, \tag{2.18}$$

$$l_{zi} + w_{zi} + h_{zi} = 1 \qquad \forall i, j \in P, \tag{2.19}$$

$$l_{xi}, l_{yi}, l_{zi}, w_{xi}, w_{yi}, w_{zi}, h_{xi}, h_{yi}, h_{zi} \in \{0, 1\} \qquad \forall i \in P,$$

$$a_{ik}, b_{ik}, c_{ik}, d_{ik}, e_{ik}, f_{ik}, s_{ij}, n_j \in \{0, 1\} \qquad \forall i, k \in P, j \in V$$

$$x_i, y_i, z_i \geq 0 \qquad \forall i \in P.$$

The objective function, as defined in equation (2.1), aims to minimize the unused volume of all vans used to pack parcels. This is done by summing the volume of the actual used vans (calculated using n_j) minus the total volume of all parcels.

Constraints (2.2)–(2.7) are essential for ensuring that the parcels do not overlap with each other. In order to explain how these constraints work, constraints (2.2) will be used as an example, but note that the same approach applies to all the other constraints as well. In constraints (2.2), the variables are bounded only when $a_{ik} = 1$. If $a_{ik} \neq 1$, then the right-hand side of the equation activates a large value M, rendering the constraint irrelevant.

When $a_{ik} = 1$, it means that the location of parcel i is to the left of parcel k. Consequently, the LBBC of parcel k can only be assigned further along the x-axis than parcel i. Therefore, x_k must be greater than x_i plus the length of parcel i in the x-axis direction. Depending on the orientation of the parcel, the length can be defined by p_i, q_i, or r_i, corresponding to the length, width, and height of the parcel respectively. Constraints (2.3)–(2.7) work in a similar way but take into account different directions and relative positions of the parcels.

Constraints (2.8) guarantee that if parcels i and k are assigned to the same van j, they must have at least one relative position to each other. This is important to avoid the possibility of the variables $a_{ik},...,f_{ik}$ being assigned a value of 0, which would make constraints (2.2)–(2.7) ineffective. To ensure that each parcel is assigned to one and only one van, constraints (2.9) are implemented. Additionally, constraints (2.10) ensure that parcels are not assigned to a van until that van has been selected. Constraints (2.11),(2.12) and (2.13) certify that the parcels packed into van j will not exceed the dimensions of the van. Finally, Constraints (2.14)–(2.19) establish that, for each parcel, its length, width and height have a one-to-one relationship with the x-, y- and z-axes.

2.2.4. Dependent variables exclusion

Examining constraints (2.14)–(2.19), it is apparent that the orientation variables are interdependent. By expressing certain variables in terms of others, it is possible to eliminate five variables, thereby significantly reducing the size of the model [8]. Specifically, the variables l_{yi}, w_{zi}, w_{xi}, h_{xi}, and h_{yi} can be eliminated. The reformulated variables are presented below:

$$\begin{aligned}
l_{yi} &= 1 - l_{xi} - l_{zi}\\
w_{zi} &= 1 - l_{zi} - h_{zi}\\
w_{xi} &= 1 - w_{yi} - w_{zi} = 1 - w_{yi} - (1 - l_{zi} - h_{zi})\\
&= 1 - w_{yi} - 1 + l_{zi} + h_{zi} = l_{zi} + h_{zi} - w_{yi}\\
h_{xi} &= 1 - l_{xi} - w_{xi} = 1 - l_{xi} - \left(1 - w_y - w_z\right)\\
&= 1 - l_{xi} - 1 + w_{yi} + w_{zi} = w_{yi} + w_{zi} - l_{xi}\\
&= w_{yi} + 1 - l_{zi} - h_{zi} - l_{zi}\\
&= 1 - l_{zi} - h_{zi} - l_{xi} + w_{yi}\\
h_{yi} &= 1 - h_{xi} - h_{zi}\\
&= 1 - \left(1 - l_{zi} - h_{zi} - l_{xi} + w_{yi}\right) - h_{zi}\\
&= 1 - 1 + l_{zi} + h_{zi} + l_{xi} - w_{yi} - h_{zi}\\
&= l_{zi} + l_{xi} - w_{yi}
\end{aligned}$$

This leads to the following reformulation of the constraints (2.2)–(2.7) and (2.11)–(2.13):

$$x_i + p_i l_{xi} + q_i\left(l_{zi} - w_{yi} + h_{zi}\right) + r_i\left(1 - l_{xi} - l_{zi} + w_{yi} - h_{zi}\right) \le x_k + (1 - a_{ik})\cdot M \quad \forall i,k : i < k \tag{2.2$'$}$$

$$x_k + p_k l_{xk} + q_k\left(l_{zk} - w_{yk} + h_{zk}\right) + r_k\left(1 - l_{xk} - l_{zk} + w_{yk} - h_{zk}\right) \le x_i + (1 - b_{ik})\cdot M \quad \forall i,k : i < k \tag{2.3$'$}$$

$$y_i + q_i w_{yi} + p_i(1 - l_{xi} - l_{zi}) + r_i\left(l_{xi} + l_{zi} - w_{yi}\right) \le y_k + (1 - c_{ik})\cdot M \quad \forall i,k : i < k \tag{2.4$'$}$$

$$y_k + q_k w_{yk} + p_k(1 - l_{xk} - l_{zk}) + r_k\left(l_{xk} + l_{zk} - w_{yk}\right) \le y_i + (1 - d_{ik})\cdot M \quad \forall i,k : i < k \tag{2.5$'$}$$

$$zi + r_i h_{zi} + q_i(1 - l_{zi} - h_{zi}) + p_i l_{zi} \le z_k + (1 - e_{ik})\cdot M \quad \forall i,k : i < k \tag{2.6$'$}$$

$$z_k + r_k h_{zk} + q_k(1 - l_{zk} - h_{zk}) + p_k l_{zk} \le z_i + \left(1 - f_{ik}\right)\cdot M \quad \forall i,k : i < k \tag{2.7$'$}$$

$$x_i + p_i l_{xi} + q_i\left(l_{zi} - w_{yi} + h_{zi}\right) + r_i\left(1 - l_{xi} - l_{zi} + w_{yi} - h_{zi}\right) \le L_j + (1 - s_{ij})\cdot M \quad \forall i, \tag{2.11$'$}$$

$$y_i + q_i w_{yi} + p_i(1 - l_{xi} - l_{zi}) + r_i\left(l_{xi} + l_{zi} - w_{yi}\right) \le W_j + (1 - s_{ij})\cdot M \quad \forall i, \tag{2.12$'$}$$

$$zi + r_i h_{zi} + q_i(1 - l_{zi} - h_{zi}) + p_i l_{zi} \le H_j + (1 - s_{ij})\cdot M \quad \forall i \tag{2.13$'$}$$

Chen *et al.* [8] conducted several tests and applications to validate the proposed model. Further information on this can be found in their paper.

2.3. Improved MILP by Pedruzzi *et al.*

Although Chen *et al.* [8] laid a solid foundation for the model, there are still some unresolved issues. In 2016, Pedruzzi *et al.* [22] improved the existing model by making some necessary changes. There are three shortcomings in the model they focus on:

1. Parcels may float in the solution.
2. The order of removal of the parcels is not considered.
3. There is no assurance for the supporting area.

The following section discusses each of these issues in detail and explores how the modification of Pedruzzi *et al.* helps to address these concerns.

2.3.1. Non-floating constraints

The first shortcoming of the model that is addressed is that the parcels are able to float. In this context, “floating" refers to a situation where the LBBC of a parcel has a z-value that is not equal to 0, but there is no parcel underneath it to support it. This does not reflect real-life situations. To correct this issue, a term is added to the objective function that minimizes the values of x_i, y_i and z_i, while multiplying it by a small parameter to ensure that it does not have a significant impact on the decision process of assigning parcels to vans. As a result, the parcels are placed in a more compact manner, which ensures that parcels are always placed as low as possible on the z-axis. This means that each parcel is either placed on the floor or on another parcel, making the model more realistic.

2.3.2. Last-In-First-Out (LIFO) constraints

Another important addition to the model is the inclusion of Last-In-First-Out (LIFO) constraints. LIFO is a term widely used in the logistics sector, which dictates that the last item placed inside van should be the first to be removed from it. To implement this, an extra parameter, $o_i, \forall i \in P$, is introduced to indicate the delivery sequence of each parcel. It is required that if $o_i < o_k$, then parcel i must be delivered before parcel k. The constraints that ensure the LIFO order are as follows: for parcels i and k, if $o_i < o_k$, then parcel i cannot be placed under or behind parcel k. This is translated into mathematical constraints in the following way:

$$x_i + (1 - c_{ik}) \cdot M \geq x_k \qquad \forall i,k \in P : i \neq k, o_i < o_k \tag{2.20}$$

$$x_i + p_i l_{xi} + q_i (1 - l_{xi}) + (1 - c_{ik}) \cdot M \geq x_k + p_k l_{xk} + q_k (1 - l_{xk}) \qquad \forall i,k \in P : i \neq k, o_i < o_k \tag{2.21}$$

$$z_i + (1 - e_{ik}) \cdot M \geq z_k \qquad \forall i,k \in P : i \neq k, o_i < o_k \tag{2.22}$$

$$z_i + r_i + (1 - e_{ik}) \cdot M \geq z_k + r_k \qquad \forall i,k \in P : i \neq k, o_i < o_k \tag{2.23}$$

If $c_{ik} = 1$, it means that parcel i is placed further back in the van than parcel k. To ensure that parcel i can still be accessed before parcel k is removed from the van, they must be placed next to each other on the x-axis. This requirement is enforced through constraints (2.20) and (2.21). Constraints (2.22) and (2.23) operate in the same manner for the z-axis.

The paper assumes that the height of each parcel should be parallel to the z-axis, which means that $h_{zi} = 1$. As a result, constraints (2.20) and (2.21) include only q_i and p_i, while constraints (2.22) and (2.23) only include r_i.

To ensure the proper functioning of these constraints, it is necessary to include relative position variables for $k < i$. This is because constraints (2.20)-(2.23) rely on the use of e_{ik} and c_{ik} for all $i,k \in P$. As a result, the following constraints are introduced:

$$a_{ik} = b_{ki}, \qquad \forall i,k \in P : i \neq k$$

$$c_{ik} = d_{ki}, \qquad \forall i,k \in P : i \neq k$$

$$e_{ik} = f_{ki}, \qquad \forall i,k \in P : i \neq k$$

For every constraint in the model, $\forall i,k : i < k$ is changed into $\forall i,k : i \neq k$.

2.3.3. Supporting area constraints

Pedruzzi *et al.* introduced constraints to ensure that every parcel in their model is sufficiently supported by either the floor or another parcel below it. They achieved this by introducing a new parameter, $as \in [0,1]$

(area support), which specifies how well each direction of the parcel should be supported. For instance, if $as = 0.8$, at least 80% of the parcel's length on the x-axis and 80% of the parcel's length on the y-axis must be in contact with either the parcel below it or the floor, resulting in 64% support of the bottom surface area. The constraints that enforce these requirements are listed below:

$$x_i + p_i l_{xi} + q_i (1 - l_{xi}) + (1 - e_{ik}) \cdot M \geq x_k \qquad \forall i,k : i \neq k \qquad (2.24)$$

$$x_i + p_i l_{xi} + q_i (1 - l_{xi}) - x_k + (1 - e_{ik}) \cdot M \geq \left(p_k l_{xk} + q_k (1 - l_{xk})\right) \cdot as \qquad \forall i,k : i \neq k \qquad (2.25)$$

$$y_i + p_i l_{yi} + q_i \left(1 - l_{yi}\right) + (1 - e_{ik}) \cdot M \geq y_k \qquad \forall i,k : i \neq k \qquad (2.26)$$

$$y_i + p_i l_{yi} + q_i \left(1 - l_{yi}\right) - y_k + (1 - e_{ik}) \cdot M \geq \left(p_k l_{yk} + q_k \left(1 - l_{yk}\right)\right) \cdot as \qquad \forall i,k : i \neq k \qquad (2.27)$$

The constraints ensure that each parcel is well supported. They apply only when parcel i is placed underneath parcel k (i.e., when $e_{ik} = 1$). In this case, the position of parcel k on the x- and y-axes must be bounded from above by the position of parcel i plus the length of parcel i on the respective axis. These bounds are enforced by constraints (2.24) and (2.26). Additionally, the difference between the endpoint of parcel i on the x-axis and the starting point of parcel k on that axis must not exceed as times the length of parcel k on the x-axis. Similar constraints are applied to the y-axis, as shown in (2.25) and (2.27). For a more elaborate explanation of the constraints, see Pedruzzi's paper [22].

2.4. Heuristic solution methods for 3D-BPP

In addition to analytical models, several heuristics have been developed to solve the problem of 3D-Bin Packing Problem (3D-BPP). This section describes the specific heuristics used in this thesis. These heuristic solve the problem as defined as in Section 1.2, so not including the constraints given by Pedruzzi [22].

2.4.1. Deepest-Bottom-Left with Fill Packing

As mentioned in Section 1.2, the Deepest-Bottom-Left with Fill (DBLF) is one of the most commonly used heuristics to solve the 3D-BPP. The pseudo-code of the heuristic can be found in Procedure 1.

Procedure 1 Deepest-Bottom-Left with Fill

```
Input: Sorted list $P$ with parcels to be packed, dimension $L$, $W$ and $H$ for the van.
Output: $x$-, $y$- and $z$-coordinates for the LBBC of every parcel
  V← $L \times W \times H$                      ▷ A list with ES sorted by non-decreasing volume
  $V' \leftarrow V$
  while $V' \neq \emptyset$ do
      while $P \neq \emptyset$ do
          Get parcel $p$ first in list $P$
          Get smallest $v$ in list $V'$
          if Parcel $p$ fits in open volume $v$ then
              check if parcel $p$ fits in one of its orientations in ES without intersecting another parcel
              place parcel $p$ in open volume $v$, minimizing first on the $x$-axis,
              then the $y$-axis and then the $z$-axis
              update the volumes of the ES in $V$
              $P \leftarrow P \setminus p$
              $V' := V$
          else
              $V' \leftarrow V' \setminus v$
          end if
      end while
      return Placement
  end while
  return False
```

The heuristic takes an ordered list of parcels as input to start the packing process. It maintains a list of all available Empty Spaces (ES) in the van, sorted in non-decreasing order of volume. An ES is a rectangular volume that is defined by two points — one closest to the origin and the other furthest from the origin — that must both fall within the dimensions of the van being used. This volume represents a space where no parcels

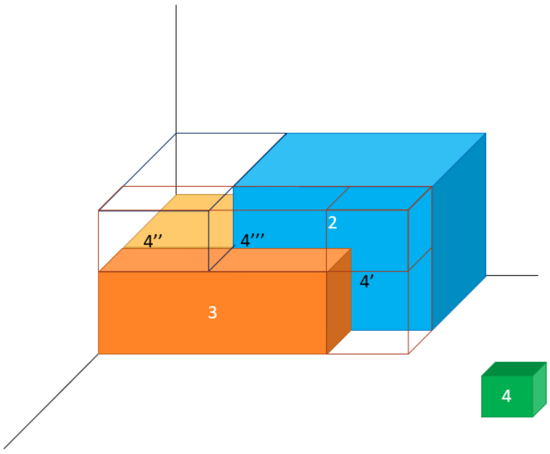


Figure 2.2: An example of DBLF. In this 3D-space three parcels have already been placed, and a fourth one still needs placing. The Empty Spaces (ES) are showed and ordered in non-decreasing from $4'$ to $4'''$.

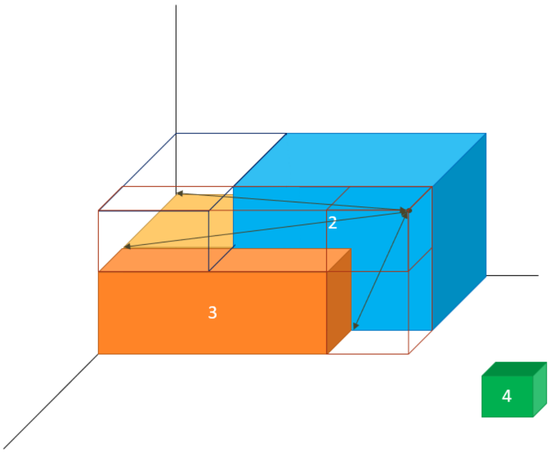


Figure 2.3: An example of DFTRC. In this 3D-space three parcels have already been placed, and a fourth one still needs placing. The Empty Spaces (ES) are showed, together with the distances to the Front-Top-Right Corner.

have been placed yet. For each parcel, the algorithm attempts to place it in the smallest ES available. If no ES has enough volume to accommodate the parcel, the algorithm returns false. Otherwise, it returns the feasible packing arrangement. The rotation of the parcels is always its length on the x-axis, width on the y-axis and height on the z-axis.

Figure 2.2 provides a visual demonstration of a single step in the algorithm. The figure depicts a van with three parcels already in place, and a fourth parcel yet to be packed. There are three ES available for the fourth parcel, ranging from $4'$ to $4'''$, sorted by volume. In this particular instance, Parcel 4 fits into the open volume $4'$, and so it is placed there accordingly.

2.4.2. Distance to the Front-Top-Right Corner

Another effective heuristic for packing parcels within a van is the Distance to the Front-Top-Right Corner (DFTRC), as proposed by Gonçalves [23]. Like the previous heuristic, this approach also involves maintaining a list of ES within the van. However, instead of prioritizing by volume, the DFTRC algorithm calculates the distance between the LBBC of the ES and the Front-Top-Right Corner of the van, and chooses the placement with the longest distance. Essentially, the same procedure as outlined in Procedure 1 can be followed, but in this case, the algorithm maintains a list of distances D, and makes its placement decisions based on these distances.

Figure 2.3 provides a visualization of the placement step in this algorithm. It shows the placement of the

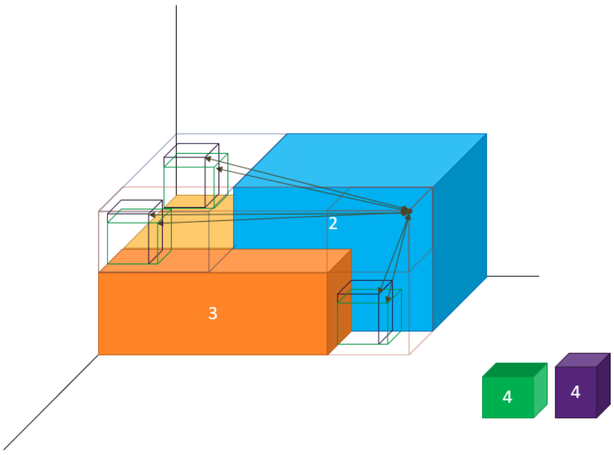


Figure 2.4: An example of DFTRC-2. In this 3D-space three parcels have already been placed, and a fourth one still needs placing. The possible placements in the Empty Spaces (ES) for both orientations is showed, together with the distances to the Front-Top-Right Corner.

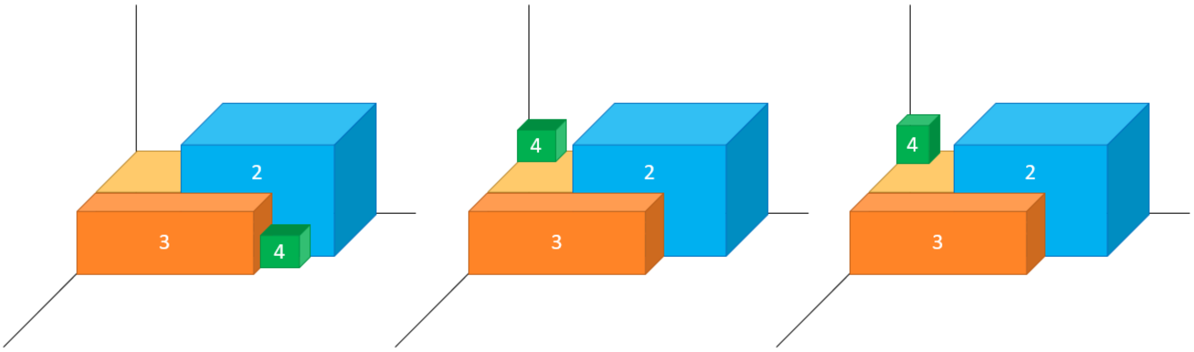


Figure 2.5: The outcome of the DBLF, DFTRC and DFTRC-2 from the examples of figures 2.2, 2.3 and 2.4, respectively from left to right.

same three boxes as in Figure 2.2, but also indicates the distance between the two previously specified points. The distance to the ES on top of parcel 1 is the longest, and therefore, this is chosen to place Parcel 4.

An improvement to the existing rule was proposed by Gonçalves [23] and is known as DFTRC-2. This rule measures the distance between the Front-Top-Right Corner of the van and the LBBC of the parcel's placements instead of the ES. This approach takes into account the possible orientations of the parcel and results in an even more compact solution. The visual representation of this approach is illustrated in Figure 2.4.

In this example, the parcel is equally long and high, and thus it can be oriented in only two ways. These different options are 'placed' into the van, and the distances are calculated. The placement of Parcel 4 on top of Parcel 1 is the furthest away, as was the case with the DFTRC. However, putting the parcel upright increases the distance to the Front-Top-Right Corner of the van, making this the better placement option.

Figure 2.5 displays the final placement achieved by all three heuristics. It is noticeable that the parcels are positioned differently for each of the three options. This illustrates that selecting a placement procedure has a significant influence on the final packing of the van.

2.4.3. Genetic Algorithm

As previously mentioned, the 3D-Bin Packing Problem (3D-BPP) problem is known to be NP-hard, meaning that finding an optimal solution can be computationally intractable for large input sizes. Although a greedy algorithm such as the Deepest-Bottom-Left with Fill (DBLF) can be used to obtain a decent solution, it may not always yield the best possible outcome. To further improve the quality of the solution, genetic algorithms can be employed to iteratively evaluate and refine the output of the heuristic.

A Genetic Algorithm (GA) is a computational technique that use principles inspired by natural selection and genetics to search for solutions to optimization and search problems [24]. The basic idea behind a GA is to start with a population of candidate solutions, each of which is represented as a string of symbols. The fitness of each individual is evaluated using a function that measures how well it performs on the problem being solved. Based on their fitness, individuals are selected for crossover to create new offspring individuals, that inherit characteristics from their parents. These offspring individuals then undergo mutation, which introduces small random changes to their genetic makeup. The process of selection, crossover, and mutation is repeated over many generations, with the purpose that the population will evolve towards better solutions to the problem.

Gonçalves [23] presents an approach to enhance the performance of the DFTRC-2 algorithm by incorporating a GA. To achieve this, Gonçalves creates a genetic string that encodes the order in which the parcels are fed into the algorithm. By evolving the genetic string using a genetic algorithm, they are able to explore and identify better orderings of the parcels, resulting in improved solutions.

3 Mixed Integer Linear Program for the 3L-BPP

In order to solve the problem as described in Section 1.1, a Mixed Integer Linear Program (MILP) is developed. This model is based on the analytical model for the 3D-Bin Packing Problem (3D-BPP) developed by Chen *et al.* [8] and the improvements made by Pedruzzi *et al.*[22], as presented in sections 2.2 and 2.3. On this basis, adjustments have been made to enable the model to solve the 3D-Bin Packing Problem with Loading Constraints (3L-BPP).

There is a need for change of the model for the following reasons: for the model solving the given problem, a feasible solution and not an optimal solution is sought. This means that there is no need for an objective function. So both minimizing the unused space as well as minimizing the x_i, y_i and z_i are being removed. Deriving from this, some extra constraints and changes are required.

The improvements made in comparison to the original model are: going from multiple vans to one, including the van interior aspects and including non-floating-, supporting area- and Last-In-First-Out constraints. All these changes are described in this chapter. The same principles to describe the van are used, so the x-axis is related to left and right, the y-axis to front and back and the z-axis for on top and under.

Analogous to the 3D-BPP, the model has two basic assumptions: Second, the dimensions of all items should be smaller than those of the loading space of the van.

3.1. Problem description

The first step in the process of creating a MILP for the problem is to define when a packing is feasible. This is done using a combination of constraints presented in other papers ([25] [19]) as well as the logical application of this problem to delivery vans. An assumption underlying the problem is that the dimensions of all items being packed into the van must be smaller than the dimensions of the van's packing space. A solution is a feasible solution if it complies with the following conditions:

1. Single van loading: It is not allowed to put parcels into a second van.
2. Non-overlapping constraints: The parcels should not overlap, which means that if a parcel is assigned a (x, y, z)-coordinate for the LBBC and a specific orientation, then no other parcel should be placed within the distance of that parcel on each axis.
3. Volumetric constraints: The dimensions of the van should be respected by the dimensions and placement of the parcels.
4. Perpendicular constraints: The parcels are not placed diagonally in the van. This means that each edge is parallel to the x-, y- or z-axis.

5. Supporting area constraints: To ensure stability, each parcel must be supported either by another parcel, the floor, or a shelf. A specified percentage of the bottom area of each parcel must be in contact with a supporting surface. It is assumed that a parcel cannot be stabilized by resting on two other parcels simultaneously.

6. Van specification input: For application to delivery vans, van-specific properties must be taken into account. These properties must be included as inputs in the model. Examples of such properties include the number of shelves in the van.

7. Last-In-First-Out (LIFO) policy: If one client is visited earlier then the others, the parcel(s) belonging to that client should be accessible from the back of the van. This can be achieved by placing the earlier parcel(s) above the other parcels, i.e., higher on the z-axis, or in front of them, i.e., further away from the origin on the y-axis. If items are placed next to each other on the x-axis, it still qualifies as a feasible LIFO loading. However, if a parcel is placed on a shelf, the LIFO order changes. For shelves on the left side of the van, parcels belonging to clients who are visited earlier should be placed further on the x-axis, while for a shelf on the right, they should be placed closer to the origin on the x-axis.

These conditions serve as the foundation for constructing the MILP.

When looking at the changes made in Section 2.2.4, it is evident that the readability is reduced. In order to restore this, new notation is introduced, namely the variables X_i, Y_i and Z_i, which are equal to the distance of parcel i on the different axes. So, the following equations are added:

$$\begin{aligned} X_i &= p_i l_{xi} + q_i \left(l_{zi} - w_{yi} + h_{zi}\right) + r_i \left(1 - l_{xi} - l_{zi} + w_{yi} - h_{zi}\right) \\ Y_i &= q_i w_{yi} + p_i \left(1 - l_{xi} - l_{zi}\right) + r_i \left(l_{xi} + l_{zi} - w_{yi}\right) \\ Z_i &= r_i h_{zi} + q_i \left(1 - l_{zi} - h_{zi}\right) + p_i l_{zi} \end{aligned}$$

3.2. Van interior modeling

This section presents constraints that precisely capture the interior of the van in the model. Initially, constraints incorporating shelves are included, followed by modifying the existing constraints to account for a single van filling.

3.2.1. Adding shelves

As delivery vans usually have shelves in their interiors, this is considered as input to the model. To incorporate the shelves in the problem, it is chosen to model them as parcels. By doing this, all the non-overlapping and volumetric constraints are incorporated at the same time as the problem is solved. The difference is that the orientation and LBBC of the shelves are already predetermined in the model, the latter denoted as $\mathscr{X}_i$, $\mathscr{Y}_i$ and $\mathscr{Z}_i$. A change is made for the input: the items to be placed inside the van are denoted as $I = P \cup S$, with P the set of parcels and S the set of shelves. This is accompanied by the the following constraints:

$$\begin{aligned} x_i &= \mathscr{X}_i && \forall i \in S \\ y_i &= \mathscr{Y}_i && \forall i \in S \\ z_i &= \mathscr{Z}_i && \forall i \in S \\ l_{xi} = w_{yi} = h_{zi} &= 1 && \forall i \in S \end{aligned}$$

3.2.2. Single van filling

Some adjustments to 3D-BPP are needed as it is designed for loading multiple vans. The parameters L_j, W_j, and H_j will become L, W, and H respectively. Additionally, the variables s_{ij} and n_j are not needed as all parcels will be placed in the same van. Hence, the constraints will be adjusted as follows:

$$\begin{aligned} a_{ik} + b_{ik} + c_{ik} + d_{ik} + e_{ik} + f_{ik} &\geq 1 && \forall i, k \in I : i \neq k, && (2.2') \\ x_i + X_i &\leq L && \forall i \in I, && (2.11') \\ y_i + Y_i &\leq W && \forall i \in I, && (2.12') \\ zi + Z_i &\leq H && \forall i \in I. && (2.13') \end{aligned}$$

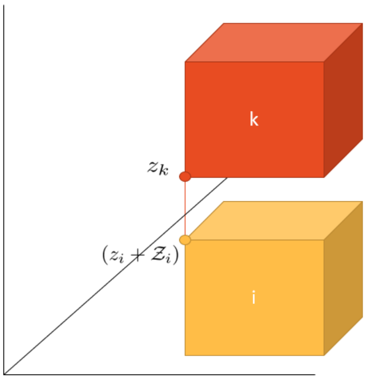


Figure 3.1: Distance between z_k and $z_i + Z_i$ of parcels i and k in a 3D-space.

3.3. Non-floating constraints

The constraints introduced in this section aim to prevent parcels from floating during loading. This is achieved by first verifying that each parcel is placed at the correct height on another parcel, and then ensuring that the underlying parcel provides enough supporting area.

3.3.1. Direct placement on another parcel

In order to prevent the parcels from floating in the van, some adjustments are made to the model. First, the floor of the van is defined as a parcel by adding it to the set S. Its LBBC is set to $(0,0,0)$, its length to L, its width to W and its height to 0. In this way it is possible to force each parcel to be directly attached to another parcel along the z-axis.

To mathematically enforce that each parcel must be directly attached to another parcel in the z-axis direction, a new binary variable n_{ik} is introduced. This variable is set to 0 if parcel k is placed directly on top of parcel i, and 1 otherwise. Additionally, a new parameter $BM \geq M + L$ is introduced for modeling purposes. Then, the following constraints are added:

$$z_k - (z_i + Z_i) + (1 - e_{ik})M \leq n_{ik} BM \qquad \forall i,k \in I : i \neq k \tag{3.1}$$

$$\sum_{i:i\neq k}^{N} n_{ik} \leq N - 1 \qquad \forall k \in P \tag{3.2}$$

$$n_{ik} \in \{0,1\} \qquad \forall i,k \in I$$

The design of the constraints is such that n_{ik} must be chosen unequal to 1 for at least one i for each parcel k. This is modelled by constraints (3.2). It can be seen that the constraints (3.1) require that if two parcels i and k are not placed on top of each other, n_{ik} must be equal to 1. This is because if $e_{ik} = 0$, the right hand side of the inequality must be greater than M, and the only way to achieve this is to set n_{ik} to 1. If $e_{ik} = 1$, this part of the inequality is equal to 0. The first part of the left hand side of the inequality represents

$$(\text{ the bottom of the upper parcel} - \text{the top of the lower parcel }),$$

which must be equal to 0 at least once, in order for the right side to also be zero. This distance is also shown in Figure 3.1. By this construction, the parcels are forced to be directly placed on at least one other parcel, and thus they are not able to float in the air. Note that shelves are allowed to float (as they are attached to the wall), and thus constraint (3.2) does not hold for shelves.

3.3.2. Supporting area constraints

Pedruzzi *et al.* [22] introduced supporting area constraints in their work to improve the stability of the packing given by the solution. In their approach, they minimized the variables x_i, y_i, and z_i to satisfy the supporting area constraints. However, this minimization is no longer used in the model, and therefore the supporting

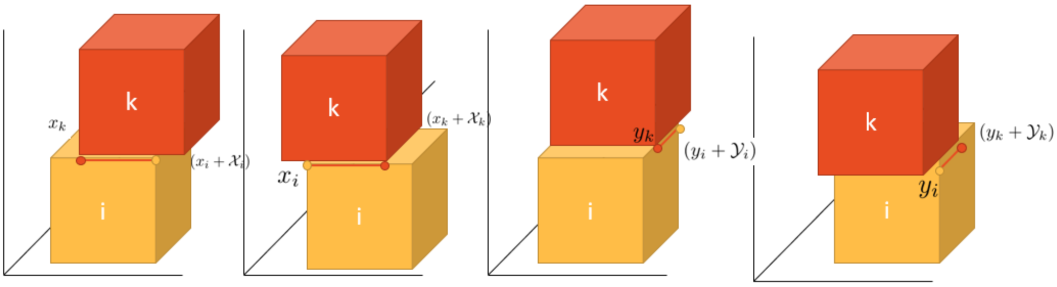
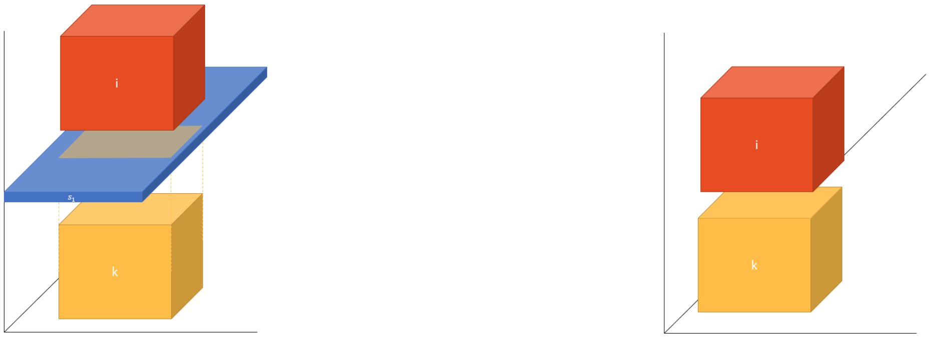


Figure 3.2: The four different distances between the corner points of two superimposed parcels i and k, which are restricted to a specific size by constraints (3.3)–(3.6).



(a) An example where parcel i is above k on a shelf, showing the restricted area.

(b) Continued example where parcel i is above k, showing that there would not be sufficient stability if the parcels were placed directly on top of each other

Figure 3.3: Example of two parcels, i and k, where i is positioned above k, but not within the restricted area outlined in constraints (3.3)–(3.6). Despite their proximity, this arrangement is permissible since they are separated by a shelf, and the variable $\lambda_{iks_1s_2}$ is introduced to account for this.

area constraints need to be reformulated.

Currently, if parcel i is positioned below parcel k, denoted by $e_{ik} = 1$, and parcel i is placed further along the x-axis than parcel k, the constraint (2.24) is still fulfilled. This same scenario applies for the three other directions and their corresponding constraints (2.25)–(2.27). As a result, it is possible for a parcel to be placed at the correct height but not above another parcel as required. To prevent this from happening, the following constraints are added:

$$(x_i + X_i) - x_k + (1 - e_{ik}) \cdot M \geq X_k \cdot as \qquad \forall i \in I, k \in P : i \neq k \tag{3.3}$$

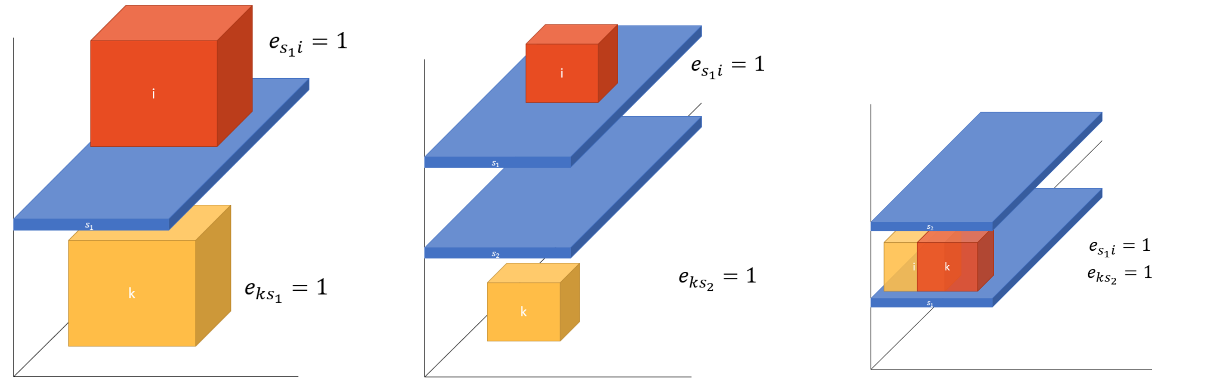
$$(x_k + X_k) - x_i + (1 - e_{ik}) \cdot M \geq X_k \cdot as \qquad \forall i \in I, k \in P : i \neq k \tag{3.4}$$

$$(y_i + Y_i) - y_k + (1 - e_{ik}) \cdot M \geq Y_k \cdot as \qquad \forall i \in I, k \in P : i \neq k \tag{3.5}$$

$$(y_k + Y_k) - y_i + (1 - e_{ik}) \cdot M \geq Y_k \cdot as \qquad \forall i \in I, k \in P : i \neq k \tag{3.6}$$

The constraints (3.3)–(3.6) use the differences between the top corners of parcel i and the bottom corners of parcel k. When $e_{ik} = 1$, this distance must be at least as times the size of parcel k on that axis. This situation is illustrated in Figure 3.2. Since constraints (3.1) and (3.2) require e_{ik} to be 1 at least once, the aforementioned constraints are met at least once, ensuring that all parcels have sufficient supporting area underneath them.

When adding constraints to restrict the placement of parcels to a specific area, it is important to consider existing constraints that affect the relative positions of parcels. There is a scenario that is not currently modeled: two parcels are partially on top of each other on separate shelves, but not completely within the constrained area. This scenario is illustrated in Figure 3.3. Due to constraints (2.2)–(2.5), a_{ik}, b_{ik}, c_{ik} and d_{ik} cannot be set to 1. In addition, the constraints (3.3)–(3.6) prevent e_{ik} and f_{ik} from being chosen as 1. To allow for this situation, a new binary variable $\lambda_{iks_1s_2}$ is introduced, which equals 1 if parcel i is on a shelf above



(a) An example where $s_1 = s_2 = s$ and $e_{ks_1} = e_{s_1 i} = 1$, resulting in $\lambda_{kiss} = 1$

(b) An example where $s_1 > s_2$ and $e_{ks_2} = e_{s_1 i} = 1$, resulting in $\lambda_{kis_1 s_2} = 1$

(c) An example where $s_1 < s_2$ and $e_{ks_2} = e_{s_1 i} = 1$, resulting in $\lambda_{kis_1 s_2} = 1$

Figure 3.4: Three different options for the variable $\lambda_{kis_1 s_2}$ getting value 1.

parcel k. To ensure that $\lambda_{iks_1 s_2}$ behaves as desired, the following constraints are added.

$$\lambda_{iks_1 s_2} \leq \frac{1}{2}(e_{s_1 i} + e_{ks_2}) \qquad \forall i, k \in P : i \neq k; \forall s_1, s_2 \in S, s_1 \geq s_2 \tag{3.7}$$

The constraints (3.7) can be divided into two cases: when $s_1 = s_2$ and when $s_1 > s_2$, where the ordering indicates that a shelf has a higher z-coordinate if it is larger. First, consider the case where $s_1 = s_2 = s$. In this case, the constraints ensure that λ_{ikss} can be set to 1 when parcel i stands on the shelf directly above k. Since $e_{si} = 1$ implies that parcel i is above shelf s, it can be deduced that parcel i is on top of shelf s. Similarly, $e_{sk} = 1$ indicates that shelf s is above k. The factor $\frac{1}{2}$ is added because λ_{ikss} can only be equal to 1 if both conditions are true. Hence, if only one of them holds, the left-hand side becomes $\frac{1}{2}$, and λ_{ikss} is then forced to become 0. An example of when $\lambda_{ikss} = 1$ can be found in Figure 3.4a.

Now say $s_1 > s_2$. These constraints are needed for cases where there are at least 2 shelves between parcels i and k. The constraints work as above, but the difference is that parcel i is on shelf s_1 and parcel k is below shelf s_2, where s_1 and s_2 are not the same shelf. An example of the situation where $\lambda_{iks_1 s_2} = 1$ can be found in the Figure 3.4b.

Please note that the variables $\lambda_{iks_1 s_2}$ are restricted to cases where $s_1 \geq s_2$. This is because, if $s_1 < s_2$, then the combination $\frac{1}{2}(e_{s_1 i} + e_{ks_2})$ would suggest that parcel i is on top of a shelf s_1 and parcel k is underneath a higher shelf s_2. This could result in violating the non-overlapping constraints after placing the parcels on the same shelf. Figure 3.4c provides an illustration of the situation that needs to be prevented.

Having established that the variables $\lambda_{kis_1 s_2}$ behave as desired, constraints (2.2) need to be adjusted. In addition to the variables $a_{ik}, \ldots, f_{ik}$, the variable $\lambda_{kis_1 s_2}$ should be included as a relative position indicator. The updated constraint is given by:

$$a_{ik} + b_{ik} + c_{ik} + d_{ik} + e_{ik} + f_{ik} \geq 1 - \sum_{\substack{s_1, s_2 \in S, \\ s_1 \geq s_2}} (\lambda_{iks_1 s_2} + \lambda_{kis_1 s_2}) \quad \forall i, k \in I : i \neq k \tag{2.2$'$}$$

3.4. Last-In-First-Out policy constraints

As explained in Section 3.1, a solution is considered feasible only if it adheres to the Last-In-First-Out (LIFO) policy. This means that for parcels i and k, if $o_i < o_k$, then parcel i cannot be placed underneath or behind parcel k. This results in the following, straightforward, constraints:

$$c_{ik} = 0 \qquad \forall i, k \in P : i \neq k; o_i < o_k \tag{3.8}$$

$$e_{ik} = 0 \qquad \forall i, k \in P : i \neq k; o_i < o_k \tag{3.9}$$

3.4.1. Parcel stacking

The constraints (3.8) do not apply if there is a shelf between two parcels, because then the order in the z-direction is reset. To take into account the effect of shelves, the variable n_{ik} is used, as it says something about being placed directly on top of each other. If n_{ik} is 0, then parcel i is directly below parcel k, and so the order should be maintained. This leads to the following adjustment of constraints (3.9):

$$e_{ik} \leq n_{ik} \qquad \forall i \in P, \forall k \in I : i \neq k; o_i < o_k \qquad (3.9')$$

3.4.2. Shelf ordering

In accordance with the problem description in Section 3.1, the order of parcel removal changes once it is placed on a shelf. Instead of being placed towards the back of the van (i.e., further along the y-axis), it should be retrieved from the end of the shelf, so closest to the line $x = \frac{L}{2}$. For the left side of the van, this means further away from the origin on the x-axis, and for the right side, it means closer to the origin. Consequently, using the c_{ik} variable is not fitting, and instead, the a_{ik} and b_{ik} variables must be used. To facilitate this, the shelves are partitioned into two groups: those on the left side and those on the right side of the van, S_l and S_r respectively, with $S = S_l \cup S_r$. The ensuing constraints are introduced to guarantee that the correct order of parcel retrieval is maintained on the shelves:

$$a_{ik} \leq (1 - e_{is}) + (1 - e_{ks}) \qquad \forall i, k \in P : i \neq k; \forall s \in S_l : o_i < o_k \qquad (3.10)$$

$$a_{ik} \leq (1 - f_{is}) + (1 - f_{ks}) \qquad \forall i, k \in P : i \neq k; \forall s \in S_l : o_i < o_k \qquad (3.11)$$

$$b_{ik} \leq (1 - e_{is}) + (1 - e_{ks}) \qquad \forall i, k \in P : i \neq k; \forall s \in S_r : o_i < o_k \qquad (3.12)$$

$$b_{ik} \leq (1 - f_{is}) + (1 - f_{ks}) \qquad \forall i, k \in P : i \neq k; \forall s \in S_r : o_i < o_k \qquad (3.13)$$

Constraints (3.10) function as follows: when parcels i and k are on the same shelf s, both e_{is} and e_{ks} are set to 1, resulting in a right-hand side of 0. The only parcels that are effected are the ones placed on the left shelf. if $o_i < o_k$, parcel k cannot be placed to the left of parcel i, so a_{ik} is forced to be 0. Constraints (3.11) enforce this for parcels on the ground beneath a shelf. Constraints (3.12) and (3.13) operate similarly, but for the right side of the van.

As e_{is} and f_{is} are now used in the model, it is necessary that these variables are only set to one if parcel i is actually on or under shelf s. As previously mentioned, constraints (3.3)–(3.6) only apply to $k \notin S$. To ensure this, similar constraints are created for both S_l and S_r:

$$X_i + x_i \leq X_s + (1 - e_{is}) \cdot M \qquad \forall i \in P; \forall s \in S_l \qquad (3.14)$$

$$X_i + x_i \leq X_s + (1 - f_{is}) \cdot M \qquad \forall i \in P; \forall s \in S_l \qquad (3.15)$$

$$x_i + (1 - e_{is}) \cdot M \geq x_s \qquad \forall i \in P; \forall s \in S_r \qquad (3.16)$$

$$x_i + (1 - f_{is}) \cdot M \geq x_s \qquad \forall i \in P; \forall s \in S_r \qquad (3.17)$$

Constraints (3.14) and (3.15) ensure that, for the shelves on the left side, the x-coordinate of the endpoint of parcel i is less than the length of the shelf. If this is the case, then parcel i is qualified as being up or under one of the shelves on the left side. Constraints (3.16) and (3.17) work similarly, with the difference that the starting point on the x-axis of parcel i should be greater than the starting point of the shelf.

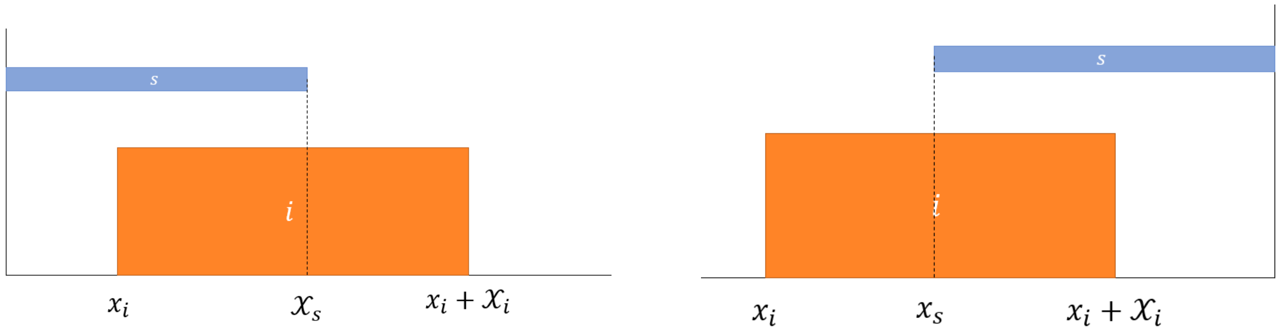
3.4.3. Aisle ordering

Lastly, a change of the constraints is required to regulate the behavior of parcels on the aisle. Firstly, a modification to (3.8) is necessary as it only applies to parcels in the aisle. If the sum of all e_{is} and f_{is} is 0 for a specific parcel i, it indicates that that parcel is in the aisle, and the original constraints should apply. This is reflected in the following constraints:

$$c_{ik} \leq \sum_{s \in S} (e_{is} + f_{is}) \qquad \forall i, k \notin S : i \neq k; o_i < o_k \qquad (3.8')$$

In addition, when a parcel is placed in the aisle, it becomes inaccessible to reach the parcels on the shelves. Therefore, constraints are needed to guarantee that the parcels placed in the aisle are the ones that need to be unloaded first from the van. This can be achieved by implementing the next constraints:

$$\sum_{s \in S} (e_{is} + f_{is}) \leq \sum_{s \in S} (e_{ks} + f_{ks}) \qquad \forall i, k \in P : i \neq k, o_i < o_k \qquad (3.18)$$



(a) Example where $\tau_{3i} = 1$. Firstly, $\tau_{1i} = 1$ because $x_i \le X_s$. Secondly, $\tau_{2i} = 1$ because $x_i + X_i \ge X_s$. This together gives the result.

(b) Example where $\tau_{6i} = 1$. Firstly, $\tau_{4i} = 1$ because $x_i \le x_s$. Secondly, $\tau_{5i} = 1$ because $x_i + X_i \ge x_s$. This together gives the result.

Figure 3.5: Examples of situations where the parcels are placed partly under the shelves. This leads to a result of $\tau_{3i} = 1$ and $\tau_{6i} = 1$.

Constraints (3.18) assure that for every parcel i, $\sum_{s \in S}(e_{is} + f_{is})$ can only become 1 if another parcel k for which yields $o_i < o_k$ already has that $\sum_{s \in S}(e_{ks} + f_{ks}) = 1$. In words that means that once a parcel is placed on the shelves, non of the successive parcel can be placed in the aisle.

3.4.4. Parcels partly under the shelf

By adding the constraints (3.14)–(3.17), it is now forbidden for parcels to be partly under a shelf and partly in the aisle. This should be allowed without re-creating the problem of misusing the e_{is} and f_{is} variables. So, new binary variables are created, τ_{3i} and τ_{6i}, which are assigned a value of 1 when a package is partially under the left or right shelf, respectively.

To ensure proper behavior of these new variables, additional binary variables τ_{1i}, τ_{2i}, τ_{4i}, and τ_{5i} have been added to reflect the position of the parcel relative to the shelf. Specifically, τ_{1i} is assigned a value of 1 if one end of the parcel is positioned under the shelves on the right, and τ_{2i} is assigned a value of 1 if the other end of the parcel is in the aisle. The same approach is applied to the other two variables. The following constraints are added to make the variables work:

$$X_s - x_i \le \tau_{1i} \cdot M, \qquad \forall i \in P, \forall s \in S_l \tag{3.19}$$

$$x_i + X_i - X_s \le \tau_{2i} \cdot M \qquad \forall i \in P, \forall s \in S_l \tag{3.20}$$

$$\tau_{1i} + \tau_{2i} \le \tau_{3i} + 1 \qquad \forall i \in P \tag{3.21}$$

$$x_s - x_i \le \tau_{4i} \cdot M \qquad \forall i \in P, \forall s \in S_r \tag{3.22}$$

$$x_i + X_i - x_s \le \tau_{5i} \cdot M \qquad \forall i \in P, \forall s \in S_r \tag{3.23}$$

$$\tau_{4i} + \tau_{5i} \le \tau_{6i} + 1 \qquad \forall i \in P \tag{3.24}$$

$$\tau_{1i}, \tau_{2i}, \tau_{3i}, \tau_{4i}, \tau_{5i}, \tau_{6i} \in \{0, 1\} \qquad \forall i \in P$$

The constraints defined in (3.19) operate as follows: if a parcel is positioned with its left end under a shelf, then $X_s - x_i > 0$. In order to satisfy the constraint, τ_{1i} must be set to 1. Figure 3.5a illustrates this scenario. If $X_s - x_i < 0$, then $\tau_{1i} \ge 0$, and the model will select 0 if possible, as the τ variables increase the model complexity, which will be explained below.

Similarly, if the end of the parcel extends beyond the shelf, constraint (3.20) ensures that if $x_i + X_i - X_s > 0$, then $\tau_{2i} = 1$. If $\tau_{1i} = \tau_{2i} = 1$, constraint (3.21) ensures that $\tau_{3i} = 1$. If only one of τ_{1i} or τ_{2i} is 1, then $\tau_{3i} \ge 0$, and the model will select 0 if possible. These rules apply to parcels positioned on the right as well, using constraints (3.22), (3.23), and (3.24). Figure 3.5b illustrates the scenario where $\tau_{6i} = 1$.

Now that the variables behave as desired, it is necessary to construct the LIFO order constraints. Since these parcels are located in both the aisle and under the shelves, both directions must be considered. Therefore, the following constraints are needed:

$$(1 - \tau_{3i}) \ge a_{ik} + c_{ik} \qquad \forall i \in P \tag{3.25}$$

$$(1 - \tau_{6i}) \ge b_{ik} + c_{ik} \qquad \forall i \in P \tag{3.26}$$

Table 3.1: Input used in the Mixed Integer Linear Program of the 3D-Bin Packing Problem with Loading Constraints.

Symbol	Description
I	Set of all items to be placed in the van
$P \subseteq I$	Set of parcels to be placed in the van
$S \subseteq I$	Set of shelves to be placed in the van
(p_i, q_i, r_i)	Parameters describing the length, width and height of parcel i
(L, W, H)	Parameters describing the length, width and height of the van
$(\mathscr{X}_i, \mathscr{Y}_i, \mathscr{Z}_i)$	Parameters desrcibing the LBBC of shelf i.
as	percentage of a parcel that should be supported by a parcel underneath it
M and BM	Large numbers for modeling purposes, $BM > M$

In constraints (3.25) and (3.26), the variables a_{ik} and b_{ik} are used for parcels on the left and right sides, respectively, as previously done in other constraints. However, there is one final aspect that needs to be modified, as the constraints (3.14)–(3.17) must account for the possibility of τ_{3i} and τ_{6i} being equal to 1. In such cases, e_{is} or f_{is} may also need to be set to 1 to ensure that the parcels maintain their relative positions with respect to the shelves. Therefore, the following changes are made to the constraints:

$$X_i + x_i \leq X_s + (1 - e_{is}) \cdot M + \tau_{3i} \cdot 2M \qquad \forall i \in P; s \in S_l \qquad (3.14')$$

$$X_i + x_i \leq X_s + (1 - f_{is}) \cdot M + \tau_{3i} \cdot 2M \qquad \forall i \in P; s \in S_l \qquad (3.15')$$

$$x_i + (1 - e_{is}) \cdot M \geq x_s - \tau_{6i} \cdot 2M \qquad \forall i \in P; s \in S_r \qquad (3.16')$$

$$x_i + (1 - f_{is}) \cdot M \geq x_s - \tau_{6i} \cdot 2M \qquad \forall i \in P; s \in S_r \qquad (3.17')$$

After conducting some tests with the model, it was discovered that the situation where parcels are partially under a shelf and partially in the aisle was only theoretically interesting. In practice, this situation rarely occurred as the optimal solution. Additionally, including the extra variables and constraints decreased the speed and therefore the quality of the model. Consequently, for the remainder of this research, it was decided not to include these variables in the model. It is assumed that a solution is feasible if all parcels are placed entirely under or on a shelf or in the aisle.

3.5. The complete Mixed Integer Linear Program (MILP)

In this section, the complete MILP is presented. Firstly, the the input parameters and variables required for the model are shown, followed by a presentation of all the constraints.

3.5.1. Input and variables

The parameters that are needed as input for the model are presented in Table 3.1 and the variables in Table 3.2.

Table 3.2: Variables used in the Mixed Integer Linear Program of the 3D-Bin Packing Problem with Loading Constraints.

Symbol	Description
(x_i, y_i, z_i)	Continuous variables indicating the (x, y, z)-coordinates of the Left-Bottom-Back Corner (LBBC) of parcel i
(X_i, Y_i, Z_i)	Continuous variables indicating the distance of parcel i on the x-, y- and z-axis
(l_{xi}, l_{yi}, l_{zi})	Binary variables indicating whether the length of parcel i is parallel to either the x-, y- or z-axis.
(w_{xi}, w_{yi}, w_{zi})	Binary variables indicating whether the width of parcel i is parallel to either the x-, y- or z-axis
(h_{xi}, h_{yi}, h_{zi})	Binary variables indicating whether the height of parcel i is parallel to either the x-, y- or z-axis
a_{ik}	1 if parcel i is on the left side of parcel k 0 otherwise
b_{ik}	1 if parcel i is on the right side of parcel k 0 otherwise
c_{ik}	1 if parcel i is behind parcel k 0 otherwise
d_{ik}	1 if parcel i is in front of parcel k 0 otherwise
e_{ik}	1 if parcel i is below parcel k 0 otherwise
f_{ik}	1 if parcel i is above parcel k 0 otherwise
$\lambda_{iks_1s_2}$	1 if parcel i is on shelf s_1 above parcel k, which is below shelf s_2 0 otherwise
n_{ik}	0 if parcel i is directly underneath parcel k 1 otherwise

3.5.2. Constraints

Now that the necessary input parameters and variables have been defined, the constraints can be presented. These constraints serve to enforce the desired conditions and restrictions on the final packing solution.

Minimize –

Subject to

$$\begin{aligned}
&x_i + X_i \leq x_k + (1 - a_{ik}) \cdot M && \forall i, k \in I : i \neq k,\\
&x_k + X_k \leq x_i + (1 - b_{ik}) \cdot M && \forall i, k \in I : i \neq k,\\
&y_i + Y_i \leq y_k + (1 - c_{ik}) \cdot M && \forall i, k \in I : i \neq k,\\
&y_k + Y_k \leq y_i + (1 - d_{ik}) \cdot M && \forall i, k \in I : i \neq k,\\
&z_i + Z_i \leq z_k + (1 - e_{ik}) \cdot M && \forall i, k \in I : i \neq k,\\
&z_k + Z_k \leq z_i + \left(1 - f_{ik}\right) \cdot M && \forall i, k \in I : i \neq k,
\end{aligned} \tag{3.27}$$

$$\begin{aligned}
&1 - \sum_{s_1, s_2 \in S} (\lambda_{iks_1s_2} + \lambda_{kis_1s_2}) \leq\\
&\qquad a_{ik} + b_{ik} + c_{ik} + d_{ik} + e_{ik} + f_{ik} && \forall i, k \in I : i \neq k,
\end{aligned} \tag{3.28}$$

$$\begin{aligned}
&x_i + X_i \leq L && \forall i \in I,\\
&y_i + Y_i \leq W && \forall i \in I,\\
&z_i + Z_i \leq H && \forall i \in I,
\end{aligned} \tag{3.29}$$

$$\begin{aligned}
&l_{xi} + l_{yi} + l_{zi} = 1, && \forall i \in I,\\
&w_{xi} + w_{yi} + w_{zi} = 1, && \forall i \in I,\\
&h_{xi} + h_{yi} + h_{zi} = 1, && \forall i \in I,\\
&l_{xi} + w_{xi} + h_{xi} = 1, && \forall i \in I,\\
&l_{yi} + w_{yi} + h_{yi} = 1, && \forall i \in I,\\
&l_{zi} + w_{zi} + h_{zi} = 1, && \forall i \in I,
\end{aligned} \tag{3.30}$$

$$\begin{aligned}
&a_{ik} = b_{ki}, && \forall i, k \in I : i \neq k,\\
&c_{ik} = d_{ki}, && \forall i, k \in I : i \neq k,\\
&e_{ik} = f_{ki}, && \forall i, k \in I : i \neq k,
\end{aligned} \tag{3.31}$$

$$\begin{aligned}
&z_k - (z_i + Z_i) + (1 - e_{ik}) \cdot M \leq n_{ik} BM && \forall i, k \in I : i \neq k,\\
&\sum_{i \in I : i \neq k}^{N} n_{ik} \leq N - 1 && \forall k \in P,
\end{aligned} \tag{3.32}$$

$$\begin{aligned}
&X_k \cdot as \leq (x_i + X_i) - x_k + (1 - e_{ik}) \cdot M && \forall i, k \in P : i \neq k,\\
&X_k \cdot as \leq (x_k + X_k) - x_i + (1 - e_{ik}) \cdot M && \forall i, k \in P : i \neq k,\\
&Y_k \cdot as \leq (y_i + Y_i) - y_k + (1 - e_{ik}) \cdot M && \forall i, k \in P : i \neq k,\\
&Y_k \cdot as \leq (y_k + Y_k) - y_i + (1 - e_{ik}) \cdot M && \forall i, k \in P : i \neq k,
\end{aligned} \tag{3.33}$$

$$\begin{aligned}
&\lambda_{iks_1s_2} \leq \frac{1}{2}(e_{s_1 i} + e_{ks_2}) && \forall i, k \in P : i \neq k;\\
& && s_1, s_2 \in S, s_1 \geq s_2,
\end{aligned} \tag{3.34}$$

$$\begin{aligned}
&X_i + x_i \leq X_s + (1 - e_{is}) \cdot M && \forall i \in P; s \in S_l, \\
&X_i + x_i \leq X_s + (1 - f_{is}) \cdot M && \forall i \in P; s \in S_l, \\
&x_s \leq x_i + (1 - e_{is}) \cdot M && \forall i \in P; s \in S_r, \\
&x_s \leq x_i + (1 - f_{is}) \cdot M && \forall i \in P; s \in S_r,
\end{aligned} \tag{3.35}$$

$$\begin{aligned}
&e_{ik} \leq n_{ik} && \forall i \in P, \forall k \in I : i \neq k : o_i < o_k, \\
&c_{ik} \leq \sum_{s \in S} (e_{is} + f_{is}) && \forall i, k \in P : i \neq k, o_i < o_k, \\
&\sum_{s \in S} (e_{is} + f_{is}) \leq \sum_{s \in S} (e_{ks} + f_{ks}) && \forall i, k \in P : i \neq k, o_i < o_k,
\end{aligned} \tag{3.36}$$

$$\begin{aligned}
&a_{ik} \leq (1 - e_{is}) + (1 - e_{ks}) && \forall i, k \in P : i \neq k, o_i < o_k; s \in S_l \\
&a_{ik} \leq (1 - f_{is}) + (1 - f_{ks}) && \forall i, k \in P : i \neq k, o_i < o_k; s \in S_l, \\
&b_{ik} \leq (1 - e_{is}) + (1 - e_{ks}) && \forall i, k \in P : i \neq k, o_i < o_k; s \in S_r, \\
&b_{ik} \leq (1 - f_{is}) + (1 - f_{ks}) && \forall i, k \in P : i \neq k, o_i < o_k; s \in S_r,
\end{aligned} \tag{3.37}$$

$$\begin{aligned}
&X_i = p_i \cdot l_{xi} + q_i \cdot w_{xi} + r_i \cdot h_{xi} && \forall i \in P, \\
&Y_i = q_i \cdot w_{yi} + p_i \cdot l_{yi} + r_i \cdot h_{yi} && \forall i \in P, \\
&Z_i = r_i \cdot h_{zi} + q_i \cdot w_{zi} + p_i \cdot l_{zi} && \forall i \in P,
\end{aligned} \tag{3.38}$$

$$\begin{aligned}
&x_i = X_i && \forall i \in S, \\
&y_i = Y_i && \forall i \in S, \\
&z_i = Z_i && \forall i \in S, \\
&l_{yi} = w_{xi} = h_{zi} = 1 && \forall i \in S.
\end{aligned} \tag{3.39}$$

$$\begin{aligned}
&l_{xi}, l_{yi}, l_{zi}, w_{xi}, w_{yi}, w_{zi}, h_{xi}, h_{yi}, h_{zi} \in \{0,1\} && \forall i \in I, \\
&a_{ik}, b_{ik}, c_{ik}, d_{ik}, e_{ik}, f_{ik}, \lambda_{iks_1s_2}, n_{ik} \in \{0,1\} && \forall i, k \in I, \\
&x_i, y_i, z_i, X_i, Y_i, Z_i \geq 0 && \forall i \in I.
\end{aligned} \tag{3.40}$$

3.6. Proof correctness of the model

In the problem description in Section 3.1, the 3D-Bin Packing Problem with Loading Constraints (3L-BPP) is defined by seven conditions (1–7). To demonstrate that the constraints outlined in this chapter result in a solution that meets the defined conditions, a two parted proof is presented. Firstly, it is shown that if a feasible packing according to the conditions is given, it satisfies all the constraints. Secondly, it is proven that if the variables are all in compliance with the constraints, it results in a feasible packing for the 3L-BPP.

Before presenting the proof, the modeling of the van will be described. The bottom left point, located as far from the back door as possible, is defined as $(0,0,0)$. On the other hand, the top right corner closest to the back door is designated as (L, W, H), representing the van's specified dimensions. All parcels will be contained within the interior of the van, occupying the three-dimensional space of $L \times W \times H$.

Lemma 1. *A feasible packing (which complies with all seven conditions as presented in Section 3.1) leads to a solution for the MILP presented in Section 3.5.*

Proof. For each parcel in the feasible packing, the corner point that is the closest to $(0,0,0)$ is the Left-Bottom-Back Corner (LBBC). We set variables $(x_i, y_i, z_i), \forall i \in P$ equal the coordinates of the LBBC. This also applies to the shelves.

For each parcel, its dimensions are specified. The longest dimension is referred to as its length p_i, the middle dimension as its width q_i, and the shortest dimension as its height r_i, so we have that $p_i \geq q_i \geq r_i$. In addition, the distances on each axis are known and set as X_i, Y_i, and Z_i. Due to condition 4, these distances will always be parallel to one of the axes.

We can establish a one-to-one relationship between X_i, Y_i, and Z_i and p_i, q_i, and r_i. This means that (X_i, Y_i, Z_i) is a permutation of (p_i, q_i, r_i), and we can find a 3×3 permutation matrix M that satisfies the equation $M \bullet (p_i, q_i, r_i) = (X_i, Y_i, Z_i)$. We set:

$$M = \begin{pmatrix} l_{xi} & w_{xi} & q_{xi} \\ l_{yi} & w_{yi} & q_{yi} \\ l_{zi} & w_{zi} & q_{zi} \end{pmatrix}$$

Then, because M is a permutation matrix, every row and column have exactly one variable equal to 1. In this way, constraints (3.30) are met.

By condition 1, we know that all parcels are placed within the same van, leading to the conclusion that every pair of parcels, i and k, have a defined relative position to one another. If they are placed on the same height, we can determine whether they are located to the left, right, front, or back of each other, and assign a_{ik}, b_{ik}, c_{ik}, or d_{ik} equal to 1 respectively. It is also important to note that if $a_{ik} = 1$, then $b_{ki} = 1$, and if $c_{ik} = 1$ then $d_{ki} = 1$, and vice versa. In all cases, we set $\lambda_{iks_1s_2} = 0$ and $n_{ik} = 1$. In the event that the parcels overlap both in the x-axis and the y-axis, meaning that they are placed above each other, there are three options:

1. They are directly placed on top of each other. Say w.l.o.g. that parcel i on top of parcel k. Then we put $f_{ik} = e_{ki} = 1$, $\lambda_{iks_1s_2} = 0$ and $n_{ki} = 0$
2. They are placed above each other on the same pile, but not directly. Say w.l.o.g. that parcel i above parcel k. Then we put $f_{ik} = e_{ki} = 1$, $\lambda_{iks_1s_2} = 0$ and $n_{ki} = 1$.
3. They are placed above each other, but there is a shelve s_1 or multiple shelves s_1 and s_2 between them ($s_1 \geq s_2$). Say w.l.o.g. that parcel i above parcel k. Then we put $f_{ik} = e_{ki} = 0$, $e_{s_1i} = e_{ks_2} = 1$, $\lambda_{iks_1s_2} = 1$ and $n_{ki} = 1$.

Now that all the variables in the model are assigned to a value, it is needed to verify that our solution satisfies all the constraints. This involves confirming that each assigned value of the variable is compatible with the constraints imposed upon it. Given conditions 1 and 3, it follows that all parcels are placed within the van, thereby ensuring that constraints (3.29) are met. Constraints (3.31), (3.38), (3.40) and (3.39) also follow directly, because of the way we choose the variables and because of condition 6. Finally, constraints (3.34) follow directly from point 3, as $\lambda_{iks_1s_1} = 1$ only when $e_{s_1i} = e_{ks_2} = 1$.

As stated earlier, every two parcels in the van have a relative position to each other. As can be seen in the way that the variables are chosen, either one of $a_{ik}, b_{ik}, c_{ik}, d_{ik}, e_{ik}, f_{ik}$ or $\lambda_{iks_1s_2}$ is 1. So, constraints (3.28) are met. To illustrate the satisfaction of constraints (3.27), we will demonstrate the validity of the first constraint, as the reasoning can be extended to the other five constraints. If $a_{ik} = 0$ the constraint is automaticly met. Now if $a_{ik} = 1$, parcel i is to the left of parcel k. Now the endpoint of parcel i, $x_i + X_i$, is before the start of parcel k, x_k, as they are non-overlapping by condition 2. Thus the constraints are satisfied.

The constraints in equation (3.33) are straightforward to verify. Only a check is needed when $e_{ik} = 1$, so parcel k is placed above parcel i. As per condition 5, every parcel is supported by another parcel or the floor/shelves, ensuring that the constraints are always met. This follows in a similar way for constraints (3.35), but then for the shelves. If $e_{is} = f_{is} = 1$, parcels are underneath or above a shelf. So their coordinates match with the wanted placement, meeting the constraints.

We know that each parcel is placed on another parcel, floor or a shelf. So, for a parcel k, we have (from point 1) that there is one parcel/floor/shelf i such that $n_{ik} = 0$. This leads to the fulfillment of the second line in constraints (3.32). The first line in these constraints only need verification if $n_{ik} = 0$. Then the left hand side should be 0 as well. The first part, $e_{ik} = 0$, follows directly from our choice in point 1. The requirement of $z_k - (z_i + Z_i) = 0$ is satisfied as the distance between the top of parcel i and the bottom of parcel k is indeed 0 if they are placed on top of each other. Thus, these constraints are also fulfilled.

Lastly, we demonstrate that constraints (3.36) and (3.37) are satisfied by using condition 7. In the first line of (3.36), certification is only needed if $n_{ik} = 0$ and $o_i < o_k$. We know by the condition that *'the earlier parcel should be placed above the other parcels (so higher on the z-axis)'*. Thus we have that $e_{ik} = 0$ if parcel i is before parcel k in the order, fulfilling the constraints.

The second line of constraints (3.36) only needs verification for parcel i if $\sum_{s\in S}(e_{is} + f_{is}) = 0$, so that means when the parcel is in the aisle. Then by condition 7, no parcel k can be placed in front of it if $o_i < o_k$, so the constraints are met. Finally, the third line only needs to be checked for parcels i and k if $\sum_{s\in S}(e_{ks} + f_{ks}) = 0$, which means that parcel k is placed on the aisle. As $o_i < o_k$, parcel i needs to be reached earlier then parcel k. So if k is placed on the aisle, by condition 7 parcel i is also placed on the aisle, and thus also $\sum_{s\in S}(e_{is} + f_{is}) = 0$ and the constraints are met. We have that $\sum_{s\in S}(e_{is} + f_{is}) \leq 1, i \in P$, so no other situations have to be verified.

To verify that all constraints (3.37) are satisfied, we first check the first line. Conformation is only needed when $e_{is} = e_{ks} = 1, s \in S_l$, indicating that parcels i and k are on the same shelf on the left side. We leverage, again by condition 7, the fact that: *'[...] for shelves on the left side of the van, that parcels belonging to clients which are visited earlier are placed further on the x-axis'*. Hence, if $o_i < o_k$, parcel i is not placed on the left of parcel k and thus $a_{ik} = 0$. Following this reasoning through for the other side, all constraints (3.37) are met.

Therefore, by using a feasible packing that adheres to all the conditions outlined in Section 3.1, we have successfully derived a solution to the model presented in Section 3.5. This completes the proof. □

Lemma 2. *A solution to the MILP presented in 3.5 results in a feasible packing solution that satisfies all seven conditions outlined in Section 3.1.*

Proof. Given the values for all variables in the MILP model presented in Section 3.5, chosen in a manner that satisfies all constraints, we will demonstrate that it corresponds to a packing that adheres to the seven conditions outlined in Section 3.1. We only need to check conditions 2–7, as condition 1 is implicit.

To do this, we construct a feasible packing by placing the LBBC of each parcel i at (x_i, y_i, z_i) in the 3D space of the van. The orientation of the parcel is determined by the variables l, w, and h. For instance, if $l_{xi} = 1$, then parcel i is aligned with its length parallel to the x-axis. The same logic applies for the other axes. By doing so, the sides of each parcel are parallel to the axes, and thus condition 4 is directly satisfied.

Condition 2 is fulfilled by constraints (3.27), which prevent parcels placed next to each other from overlapping in their coordinates. For a more detailed explanation, please refer to Section 2.2.3. In addition, condition 3 is satisfied by constraints (3.29), as these ensure that all parcels are placed inside the van while respecting its dimensions.

Condition 5 is achieved through the combined implementation of constraints (3.32) and (3.33). Constraints (3.32) mandate that each parcel is placed on either another parcel, the floor, or a shelf. Constraints (3.33) then ensure that, if a parcel is placed on another parcel, it is adequately supported. A more in-depth examination of these constraints and how they function can be found in Section 3.3.2. Furthermore, condition 6 is satisfied by using the variables in constraints (3.39) as the shelves within the van.

Condition 7 is satisfied through the interplay of constraints (3.36) and (3.37). The first line of (3.36) ensures a proper ordering in the z-axis, while the second line establishes the proper order towards the door for parcels in the aisle. The third line ensures that parcels can only be placed in the aisle once all preceding parcels have also been placed there. For a more comprehensive explanation of these constraints, see Section 3.4. In the same section, the role of constraints (3.37) in establishing the correct order on shelves is also ex-

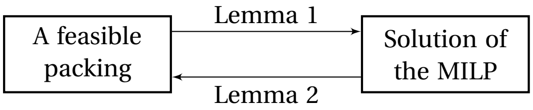


Figure 3.6: Proof structure for Theorem 3

plained.

So, by adhering to all the constraints outlined in the MILP in Section 3.5, we have established that the resulting packing meets all conditions (1)–(7) as described in Section 3.1. This concludes the proof. □

Theorem 3. *The MILP in Section 3.5 correctly models the 3D-Bin Packing Problem with Loading Constraints (3L-BPP).*

Proof. This immediately follows from the combination of Lemmas 1 and 2. This is made visual in Figure 3.6. □

4

Heuristic Solution Method for the 3L-BPP

This chapter presents a heuristic solution method developed for the 3D-Bin Packing Problem with Loading Constraints (3L-BPP), based on the DFTRC-2 described in Section 2.4.2. The adjustments to the basic method aim to make it more applicable to the 3L-BPP and produce an initial solution for the Mixed Integer Linear Program (MILP) (presented in Chapter 3). The heuristic method starts by addressing the floating parcel issue, then incorporates shelves into the van, and finally modifies the distance measurement method.

4.1. Non-floating improvements

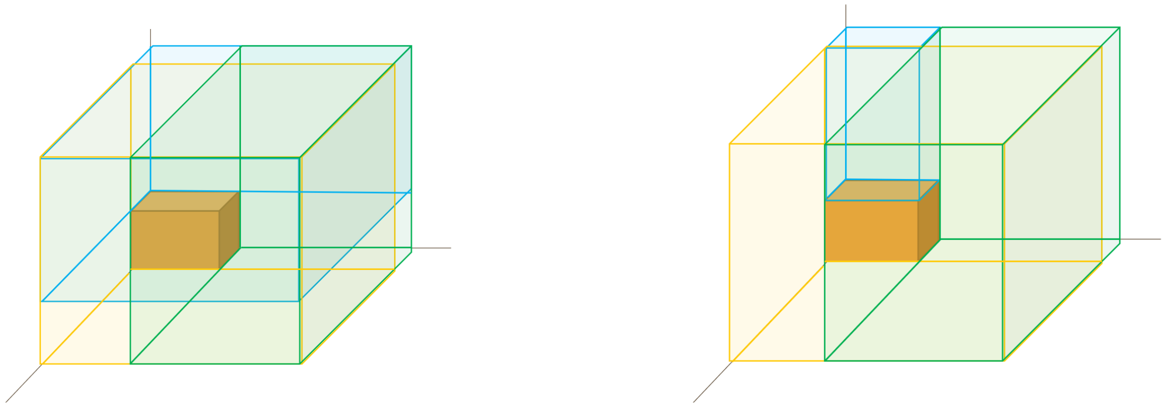
When looking at the DFTRC-2, it can be seen that the x-, y- and z-axis are all treated the same. although this is logical from an algorithmic point of view, this does not solve the problem as described in Section 3.1. By treating all the axes the same, the parcels can 'float', meaning they do not have any other parcel, a shelf or the floor underneath it. This is because of how the Empty Spaces (ES) are created. If a parcel is placed, three new ES are created; one on each axis along its sides. This is made visible in Figure 4.1a, where every newly created ES is showed by a different color. This allows for the placement of parcels in the resulting ES on the z-axis, but without stability guaranteed by another parcel. This problem can be solved by changing the ES from the whole area above the parcel, to only the area directly adjacent to the parcel itself. This change can be seen in figure 4.1b.

There is one other change needed to ensure the stability of the parcels. For the parcels that are placed on the ground, the ES are now created in a correct way. But once a second parcel is placed on top of it, new issues occur. That is because the ES in the direction of the x-axis and y-axis are going all the way to the back of the van. Because these areas are already above the ground, this creates ES where no stability on the parcels is ensured. This issue is tackled by changing these ES in such a way that they stop at the x- and y-coordinate of the parcel below. In this way, it is still ensured that every parcel is standing on the floor or on top of another parcel. The first situation is visible in Figure 4.2a and the situation after the change can be seen in Figure 4.2b.

4.2. Adding shelves

For the original model, as elaborated in Section 2.4.2, this problem is solved in an empty squared space. The 3L-BPP, however, involves parcel delivery vans that may have shelves. To address this, a modification to the method is proposed. The initial ES is the space starting at $(0,0,0)$ and ending in (L,W,H). This one is also shown in Figure 4.3a. By changing this initial ES, shelves can be added.

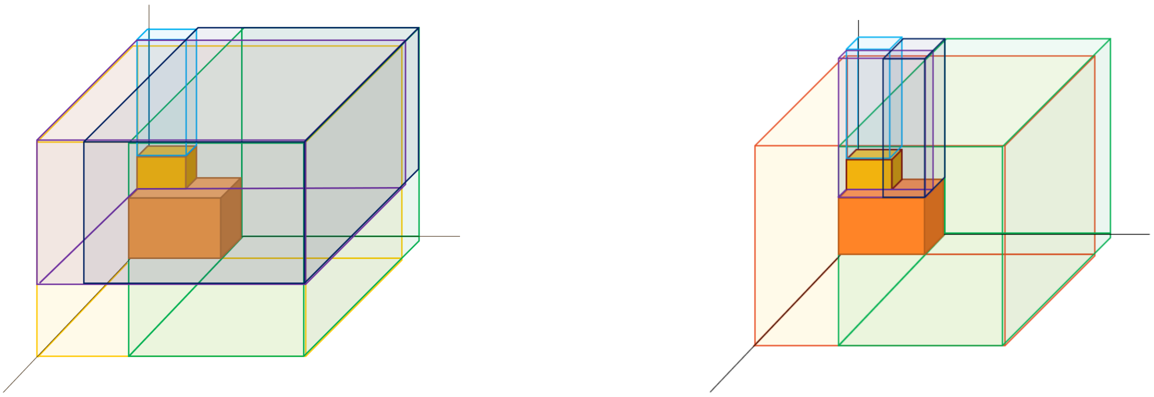
For example, if $|S| = 2$, five ES will be created; two below the shelves, two above the shelves and one between the shelves. This situation can be seen in Figure 4.3b. Using this as the initial ES, the method will be able to place parcels here and change the empty spaces to smaller ones as soon as a parcel is placed there, but will not create any new ES in the air. So this, combined with the changes proposed in Section 4.1, the method will always place a parcel on the ground, on another parcel, or on a shelf. The shelves can thus be taken as input with this change, as other input will produce other initial ES, but will not change the method itself.



(a) The original method for constructing the ES, which involves creating three 3D-spaces towards the corner point (L, W, H).

(b) The adjusted method for constructing the ES, which involves creating a 3D-space above the parcel that extends towards the corner point located at the Right-Front of the parcel, and has a height of H.

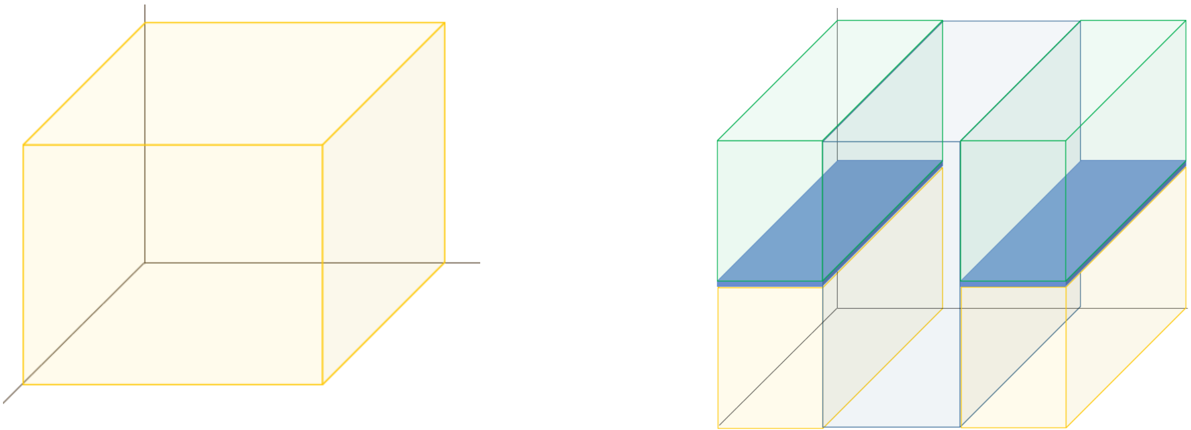
Figure 4.1: Example of how the heuristic generates Empty Spaces (ES) for a first parcel and adjustments to this process.



(a) The method for constructing the ES for a second parcel, which involves creating two 3D-spaces towards the point (L, W, H)

(b) The adjusted method for constructing the ES for a second parcel, which involves creating three 3D-spaces around the parcel that extends towards the corner point located at the Right-Front of the parcel, and has a height of H.

Figure 4.2: Example of how the heuristic creates Empty Spaces (ES) for a second parcel and adjustments to this creation.



(a) The initial ES when no shelves are used.

(b) The initial ES when two shelves are used.

Figure 4.3: Example of two shelves are used as input to change the initial ES for the heuristic.

4.3. Distance determination methods

The original heuristic solution method aimed to place parcels as compactly as possible in a quick manner. However, for the 3L-BPP, an additional objective has been introduced: to adhere to the Last-In-First-Out (LIFO) order of the parcels. This greedy solution method operates within polynomial time and, unless $NP = P$ is proven, it may not always provide an optimal solution. Nevertheless, by altering the placement method, it can yield a more coherent solution with respect to the LIFO order. One way to achieve this is to modify the DFTRC method with a different distance determination.

4.3.1. Three enhancements to distance determination

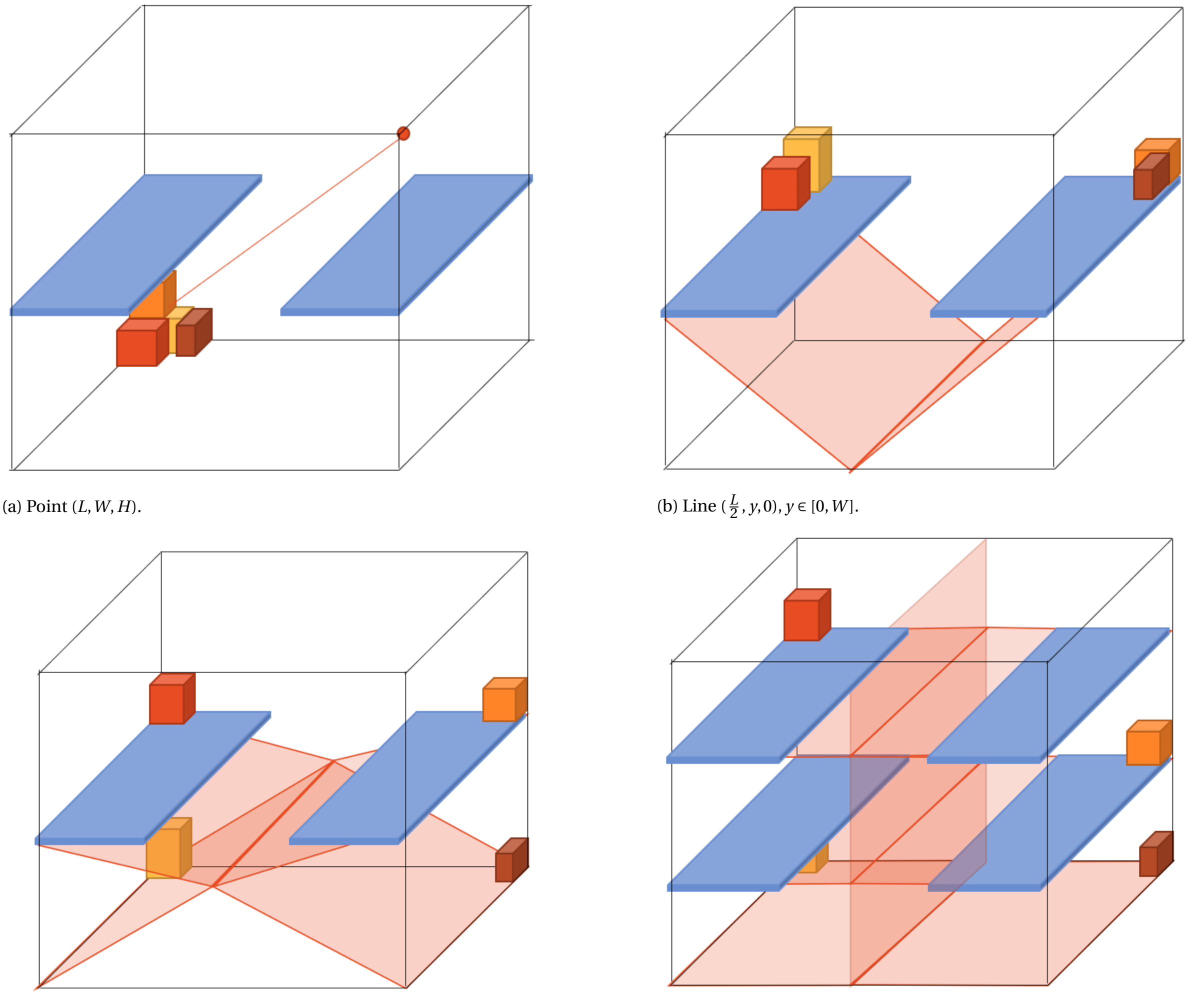
Figure 4.4 presents various options for distance determination. In Figure 4.4a, the method (DFTRC) outlined in Section 2.4.2 is used, which involves computing the maximal distance to the point (L, W, H). This results in a tightly packed configuration in the lower left corner. To calculate the placement of parcel i, denoted as ρ_i, the following formula is employed:

$$\begin{aligned}\rho_i = \max\{&(L - x_i - X_i)^2 + (W - y_i - Y_i)^2 + (H - z_i - Z_i)^2;\\ &(x_i, y_i, z_i) \text{ the LBBC for every ES;}\\ &X_i, Y_i \text{ and } Z_i \text{ for all possible orientations}\}\end{aligned}$$

Where X_i, Y_i and Z_i are taken as in Chapter 3, i.e. the distance of the parcel on the x-, y- and z-axis respectively. This is checked for all six different orientations of the parcel and the placement for which this sum is the largest, is taken.

The results obtained using a different distance formula are shown in Figure 4.4b. This formula was developed by taking into account the LIFO condition for the model described in Section 3.1. According to this condition, it is not desirable to place parcels that are going out last on the aisle. Additionally, the parcels on the shelves should be arranged in the order of their x-axis position (i.e., length of the van), from back to front. However, it is not necessary for the parcels to be ordered from back to front on the y-axis. Therefore, the distance to the y-axis can be disregarded, and a line is selected as the reference point for distance measurement. Ideally, the floor should be left unused, so the farthest distance to the ground (which is at height $z = 0$) is chosen as the base for measuring distance. Consequently, the maximal distance between the ES and the line $(\frac{L}{2}, y, 0), y \in [0, W]$ is calculated.

When the distance is measured between the line $(\frac{L}{2}, y, 0), y \in [0, W]$ and the LBBC of the ES, it creates inconsistencies in the measurements. This is since the left corner of the parcel is chosen. If choosing between placing a parcel with its furthest corner at point $x_i = 0$ or $x_i = L$, the distance is not the same (excluding if $X_i = 0$). To address this issue, a different measurement point will be used for locations where $x_i > \frac{L}{2}$. In this case, the distance between the line and the Right-Bottom-Back Corner (RBBC) will be measured. The formula



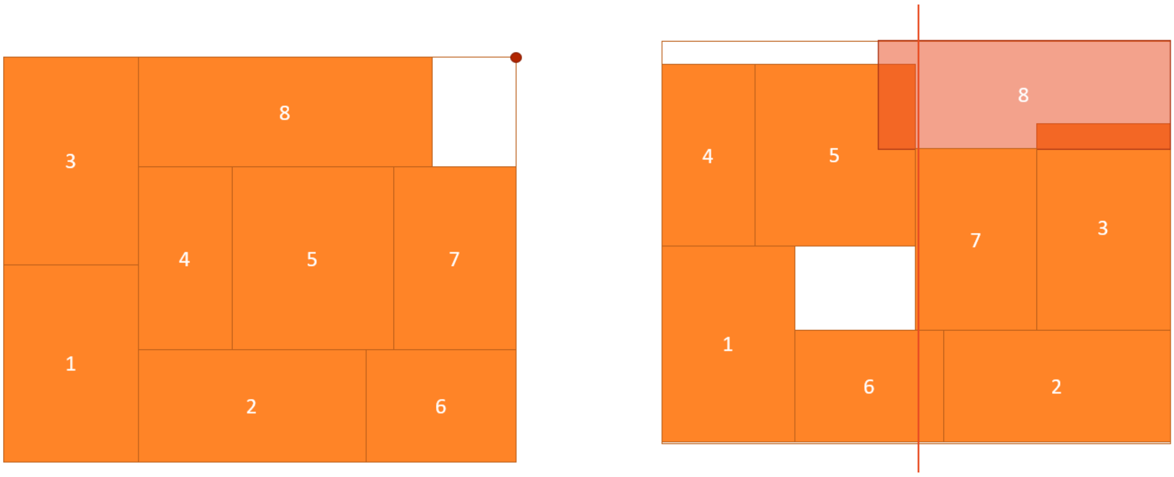
(a) Point (L, W, H).

(b) Line $(\frac{L}{2}, y, 0)$, $y \in [0, W]$.

(c) Line $(\frac{L}{2}, y, \frac{H}{4})$, $y \in [0, W]$.

(d) Face $(\frac{L}{2}, y, z)$, $y \in [0, W]$, $z \in [0, H]$.

Figure 4.4: Options for selecting a base point, line or face to determine the placement of parcels. The parcels are placed the furthest away (in euclidean distance) from this base.



(a) Point (L, W).

(b) Line $(\frac{L}{2}, y), y \in [0, W]$.

Figure 4.5: Options for selecting a base point or line to determine the farthest placement of squares in a 2-dimensional example.

for ρ_i then becomes:

$$\begin{aligned}\rho_i = \max\Bigg\{ & \left(\left(\frac{L}{2} - x_i - X_i\right)^2 + (0 - z_i - Z_i)^2 ; (x_i, y_i, z_i) \text{ the LBBC for every ES; with } x_i \leq \frac{L}{2}\right.\\ & \left. X_i, Y_i \text{ and } Z_i \text{ for all possible orientations}\right),\\ & \left(\left(\frac{L}{2} - x_i - X_i\right)^2 + (0 - z_i - Z_i)^2 ; (x_i, y_i, z_i) \text{ the RBBC for every ES with } x_i > \frac{L}{2};\right.\\ & \left. X_i, Y_i \text{ and } Z_i \text{ for all possible orientations}\right)\Bigg\}\end{aligned}$$

To ensure that the parcels are placed without preference for above or below the shelve, the base line from which the distance is taken can be chosen in between the shelves. For instance, if two shelves are placed at a height of $\frac{H}{2}$, then the baseline can be set at a height of $\frac{H}{4}$. This means that the formula used to calculate distances will remain the same as before, but with the substitution of $(0 - z_i - Zi)^2$ with $(\frac{H}{4} - z_i - Zi)^2$. An illustrative example of this can be found in Figure 4.4c.

One way to achieve a more general placement of parcels is to exclude the height variable from the equation. This method ensures that the parcels are placed without any preference for the y- and z-axes, and as far away as possible from the center of the van, while forming a face with dimensions of W by H placed at $\frac{L}{2}$ on the x-axis. This approach is particularly useful when dealing with more than two shelves ($|S| > 2$). An illustration of this placement strategy is presented in Figure 4.4d.

4.3.2. Possible compactness loss

One potential disadvantage of modifying the distance calculations is that it can reduce the compactness of the solution. This is because when the method builds up the solution from two opposite sides, gaps may appear where the two sides meet in the middle. If the parcels are placed from a single corner, the likelihood of creating gaps will be minimal. The efficacy of this approach is dependent on the input, but generally speaking, using a single corner as the distance base tends to outperform using a face or line in terms of compactness.

The problem becomes apparent when examining a 2-dimensional example. Figure 4.5 illustrates the placement of 8 squares in a square area of $L \times W$. In Figure 4.5a, the placement method maximizes the distance to the point (L, W). As shown, the squares are carefully placed either above or beside each other. Even after placing square 8, there is still some space available in the $L \times W$ square. In contrast, Figure 4.5b presents a placement method that maximizes the distance to the line $(\frac{L}{2}, y), y \in [0, W]$. Here, the squares are placed one by one on either the left or right side of the square. After square 6 is placed, square 7 cannot fit between squares 5 and 6, forcing it to be positioned to the right and leaving a significant gap between them. As a result, there is not enough contiguous area left to place square 8. This example clearly demonstrates that the solution achieved with this placement method is less compactly designed.

In developing the heuristic solution, it is necessary to balance the trade-off between maintaining the LIFO order and possibly losing some compactness in the solution. These trade-offs are based on the findings presented in Chapter 6.

5

Combined model for 3L-BPP

In this chapter, a combined model for the 3D-Bin Packing Problem with Loading Constraints (3L-BPP) is presented, which consists of the Mixed Integer Linear Program (MILP) presented in Chapter 3 and the heuristic presented in Chapter 4. When looking at the models, they both have different assets. The MILP provides a complete solution to the problem, addressing all the complexities that arise in real-world scenarios. However, it generates a large number of variables and constraints, which may result in significant runtime. On the other hand, the heuristic approach is fast, but it gives no guarantees in how well the Last-In-First-Out (LIFO) order is maintained in the solution.

The respective strengths of the models can be used in a combined approach. To do so, modifications are made to the MILP to increase its usability, even for larger datasets. Then, the heuristic method is used to find a feasible solution, with the addition of different distance base points and a Genetic Algorithm. In the last section, the two models are combined into the full model, which is utilized to gather results in Chapter 6.

5.1. Adjustments to the MILP

The MILP presented in Chapter 3 provides a feasible solution for the 3L-BPP by ordering the parcels in a way that they are reachable when visiting the client (the LIFO order). The constraints related to this order increase the complexity of the problem. To address this, adjustments are made to the model to enhance its feasibility. Firstly, improvements in efficiency are made, followed by converting these constraints to soft constraints to ensure that a solution, although not necessarily optimal, is available for any input.

5.1.1. Efficiency improvement

To improve the efficiency in terms of runtime, it is crucial to wisely choose the value of big M in the constraints. Constraints containing a big M have more expansive solution space for the linear relaxation, and decreasing this space can decrease the solving time. First, the constraints (3.27) are reviewed. When examining the dimensions on different axes, it is apparent that they can never exceed the size of the van. Therefore, the constraints are adjusted using L, W, and H. The same approach is applied to constraints (3.35). Additionally, constraints (3.32) feature both big M and BM. For big M, H can be used as these constraints manage the coordinates on the z-axis. Both parts of the left-hand side can have a maximum of H, so the right-hand side can be limited to $2H$. The modified constraints are listed below:

$$\begin{aligned}
& x_i + X_i \leq x_k + (1 - a_{ik}) \cdot L && \forall i,k \in I : i \neq k, \\
& x_k + X_k \leq x_i + (1 - b_{ik}) \cdot L && \forall i,k \in I : i \neq k, \\
& y_i + Y_i \leq y_k + (1 - c_{ik}) \cdot W && \forall i,k \in I : i \neq k, \\
& y_k + Y_k \leq y_i + (1 - d_{ik}) \cdot W && \forall i,k \in I : i \neq k, \qquad (3.27') \\
& z_i + Z_i \leq z_k + (1 - e_{ik}) \cdot H && \forall i,k \in I : i \neq k, \\
& z_k + Z_k \leq z_i + \left(1 - f_{ik}\right) \cdot H && \forall i,k \in I : i \neq k,
\end{aligned}$$

$$
\begin{aligned}
&X_k \cdot as \leq (x_i + X_i) - x_k + (1 - e_{ik}) \cdot H && \forall i,k \in P : i \neq k, \\
&X_k \cdot as \leq (x_k + X_k) - x_i + (1 - e_{ik}) \cdot H && \forall i,k \in I : i \neq k, \\
&Y_k \cdot as \leq (y_i + Y_i) - y_k + (1 - e_{ik}) \cdot H && \forall i,k \in I : i \neq k, \qquad (3.35') \\
&Y_k \cdot as \leq (y_k + Y_k) - y_i + (1 - e_{ik}) \cdot H && \forall i,k \in I : i \neq k, \\
&z_k - (z_i + Z_i) + (1 - e_{ik}) \cdot H \leq n_{ik} \cdot 2H && \forall i,k : i \neq k,
\end{aligned}
$$

Some evident constraints can be added to limit the solution space of the relaxation as well. First of all, one can see that if a parcel is on the left of another parcel, it cannot be also on the right of that parcel. This leads to the constraints $a_{ik} + b_{ik} \leq 1$. This can be extended to the other two dimensions as well. Using constraints (3.30) one more constraint can be added. One can see that for every parcel i, exactly three variables regarding the rotation have to be chosen. Adding this as a constraint also reduces the solution space without changing the model itself. The newly added constraints will be:

$$a_{ik} + b_{ik} \leq 1 \qquad \forall i,k \in I : i \neq k \qquad (5.1)$$

$$c_{ik} + d_{ik} \leq 1 \qquad \forall i,k \in I : i \neq k \qquad (5.2)$$

$$e_{ik} + f_{ik} \leq 1 \qquad \forall i,k \in I : i \neq k \qquad (5.3)$$

$$l_{xi} + l_{yi} + l_{zi} + w_{xi} + w_{yi} + w_{zi} + h_{xi} + h_{yi} + h_{zi} = 3 \qquad \forall i \in I \qquad (5.4)$$

To improve efficiency, constraints (3.36) can be modified. Currently, each constraint checks whether none of the preceding parcels have been placed on the floor. However, it suffices to only check if the directly preceding parcel is not on the ground, and then iteratively check all the previous parcels. By doing so, a reduction in the number of constraints from $(P \setminus S) \times (P \setminus S)$ to $(P \setminus S)$ is achieved. Less constraints can lead to less solving time, although this varies per model. The modified constraints are as follows:

$$\sum_{s \in S} (e_{is} + f_{is}) \leq \sum_{s \in S} (e_{(i+1)s} + f_{(i+1)s}) \qquad \forall i \in P \qquad (3.36')$$

In many MILP problems, symmetries can arise in the solution space. Multiple solutions can lead to the same objective. However, in the case of the 3L-BPP, all solutions are unique. This is because every interior of a van is different, meaning that flipping the solution as a whole has no effect.

5.1.2. Soft constraints

As mentioned earlier, the problem at hand is classified as NP-hard, meaning that finding a solution can be time-consuming. Thus, it may be advantageous to find a near-feasible solution and employ optimization techniques to reach a feasible solution. To guarantee that the model can produce a solution within a predetermined time frame, certain hard constraints are transformed into soft constraints. By allowing these constraints to be violated with an assigned penalty cost, the objective function is adjusted accordingly [26].

To achieve the desired outcome, it is chosen to revise all constraints related to the Last-In-First-Out (LIFO) order of the parcels, that include constraints (3.36) and (3.37). Upon examination, these constraints all behave the same: they enforce a binary variable to be 0 in specific scenarios. To soften these constraints, one strategy can be applied to all constraints: add a binary slack variable to the right-hand side, which can be chosen as one, and include this variable in the objective function.

To reduce the number of variables used, it is necessary to identify which situations cannot occur simultaneously. In such cases, the same variable can be used. For instance, within all the constraints (3.37), none of these can occur at the same time, as $e_{is} + f_{is} \leq 1$. Therefore, the first two and the second two can never both be true. Additionally, $e_{is_1} + e_{is_2} \leq 1 : s_1 \in S_r; s_2 \in S_l, i \in P$, as a parcel cannot be underneath a shelf on both the left and right side. So for all these situations, a binary variable $\alpha_{ik} \in \{0,1\}$ is introduced and added to the right-hand side of the constraints.

$$
\begin{aligned}
&a_{ik} \leq (1 - e_{is}) + (1 - e_{ks}) + \alpha_{ik} && \forall i,k \in P : i \neq k, o_i < o_k; s \in S_l \\
&a_{ik} \leq (1 - f_{is}) + (1 - f_{ks}) + \alpha_{ik} && \forall i,k \in P : i \neq k, o_i < o_k; s \in S_l \\
&b_{ik} \leq (1 - e_{is}) + (1 - e_{ks}) + \alpha_{ik} && \forall i,k \in P : i \neq k, o_i < o_k; s \in S_r \qquad (3.37') \\
&b_{ik} \leq (1 - f_{is}) + (1 - f_{ks}) + \alpha_{ik} && \forall i,k \in P : i \neq k, o_i < o_k; s \in S_r
\end{aligned}
$$

When examining constraints (3.36), there are three distinct scenarios. The first line pertains to the stacking order of parcels, and is relevant in all situations, regardless of whether the parcels are located on shelves or in the aisle. To accommodate this, a dedicated binary variable, $\beta_{ik} \in \{0,1\}$, is introduced and included on the right-hand side of the constraint.

The second line of constraints (3.36) only applies to parcels located in the aisle, while constraints (3.37) only apply to parcels on the shelves, and therefore, these constraints are mutually exclusive. Consequently, the binary variable α_{ik} can be used to represent both sets of constraints. Since the last line of constraints (3.36) applies to all parcels, a new binary variable is introduced, $\gamma_{ik} \in \{0,1\}$. These variables are included in the constraints, which can be seen below:

$$\begin{aligned} e_{ik} &\leq n_{ik} + \beta_{ik} && \forall i \in P, \forall k \in I : i \neq k : o_i < o_k \\ c_{ik} &\leq \sum_{s\in S}(e_{is} + f_{is}) + \alpha_{ik} && \forall i,k \in P : i \neq k, o_i < o_k \\ \sum_{s\in S}(e_{is} + f_{is}) &\leq \sum_{s\in S}(e_{(i+1)s} + f_{(i+1)s}) + \gamma_{ik} && \forall i \in P \end{aligned} \tag{3.36'}$$

The next step is to include the variables in the objective function. As there was no objective function established previously, the new objective function will consist entirely of the violation variables. The objective is to minimize the number of incorrectly placed parcels, taking all types of violations into account. When a parcel is incorrectly placed, a counter of wrongly placed parcels is set to 1. To accomplish this, an additional variable, $\delta_i \in \{0,1\}$, is introduced. The following constraints are then added to ensure that δ_i counts wrongly placed parcels correctly:

$$\delta_i \geq \alpha_{ik} \qquad \forall i,k \in P \tag{5.5}$$

$$\delta_i \geq \beta_{ik} \qquad \forall i,k \in P \tag{5.6}$$

$$\delta_i \geq \gamma_{ik} \qquad \forall i,k \in P \tag{5.7}$$

Now, the objective function value becomes as follows:

$$\min \sum_{i\in P} \delta_i \tag{5.8}$$

5.2. Adjustments to the Heuristic solution method

In this section, modifications made to the heuristic solution method (DFTRC-2) will be discussed. These modifications aim to provide feasible solutions to the MILP problem while considering the adjustments made to the model in Section 5.1.2. The adjustments include selecting a suitable distance baseline, examining the input order for the heuristic, and integrating a Genetic Algorithm (GA) into the solution method.

5.2.1. Choice for distance baseline

As discussed in Section 4.3, the choice of face, line, or point from which the distance is measured in the heuristic method has a significant impact on the solution. After conducting several tests on some try-out data, it is found that using the face $(\frac{L}{2}, y, z)$ with $y \in [0, W]$ and $z \in [0, H]$ resulted in favorable outcomes. This approach effectively exploited the sides of the van, the shelves, and the ground while minimizing the use of the aisle. However, it is found that while this approach was effective, it lacked in terms of compactness. When optimizing the distance to the middle, all the parcels were turned with the shortest side on the x-axis, resulting in a sub-optimal solution in terms of stability.

To address this issue, a minor factor is introduced that minimized the y-coordinate in addition to the optimization of the distance to the middle. This approach allowed the shelves to remain the preferred location for the parcels while choosing a more compact orientation. Specifically, the formula for placement ρ_i of

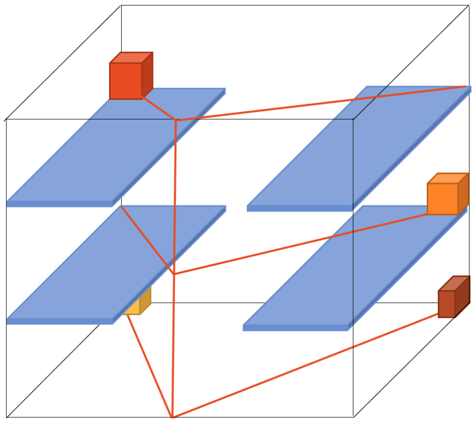


Figure 5.1: Using the line $(\frac{L}{2}, W, z), z \in [0, H]$ to determine the placement of parcels. The parcels are placed the furthest away (in euclidean distance) from this base.

parcel i is now determined as follows, with $0 < \varepsilon < 0.5$:

$$\begin{aligned}\rho_i = \max\Bigg\{ & \Bigg(\Bigg(\frac{L}{2} - x_i - X_i\Bigg)^2 + \varepsilon \cdot \big(W - y_i - Y_i\big)^2 ; \big(x_i, y_i, z_i\big) \text{ the LBBC over all the ES; with } x_i \leq \frac{L}{2} \\ & X_i, Y_i \text{ and } Z_i \text{ for all possible orientations}\Bigg), \\ & \Bigg(\Bigg(\frac{L}{2} - x_i - X_i\Bigg)^2 + \varepsilon \cdot \big(W - y_i - Y_i\big)^2 ; \big(x_i, y_i, z_i\big) \text{ the RBBC over all the ES with } x_i > \frac{L}{2}; \\ & X_i, Y_i \text{ and } Z_i \text{ for all possible orientations}\Bigg)\Bigg\}\end{aligned}$$

As a result, the line taken as the distance baseline is $(\frac{L}{2}, W, z), z \in [0, H]$. This can be seen in figure 5.1.

5.2.2. Choice for input order

Section 2.4.1 specifies that the method requires a sorted list of parcels as input, and the choice of this list has a significant impact on the outcome of the packing. To achieve a better solution in terms of maintaining the LIFO order, a smart choice must be made. The most obvious choice is to sort the parcels in reverse order of when they should be reached. This ensures that the first parcel placed is the one that needs to be reached last. Since the heuristic then places the next parcel mostly before or on top of it, the LIFO order is will be quite well maintained, resulting in favorable outcomes. Only if one of the Empty Spaces (ES) is relatively small, it is possible that many parcels may not be placed in that ES, but instead in front of it. In such cases, when a parcel that fits inside this smaller space, it will be placed behind the other parcels, which may result in minor errors in the placement concerning the LIFO order.

To address this issue, one possible solution is to modify the minimum volume of the ES. By default, this is set to the smallest volume of the remaining parcels to be packed. Increasing this minimum value will eliminate the ES more quickly, resulting in fewer ES behind the other parcels. However, this could lead to less compact packing, and the heuristic may be unable to discover a feasible packing.

5.2.3. Genetic Algorithm

Using the reverse order as input for the heuristic is a useful technique for maintaining the LIFO order. However, in some cases, this method may not result in a solution where all parcels fit inside the van. In such situations, it is chosen to adjust the order in which parcels are inputted. To accomplish this in a structured manner, a Genetic Algorithm (GA) (as detailed in Section 2.4.3) has been integrated into the heuristic solution method. The heuristic terminates once a solution is found where all parcels fit inside the van.

The initial solution of the GA is the same as the solution presented above, with parcels inputted in inverted order. Afterwards, the GA applies three operators: selection, crossover and mutation. For crossover,

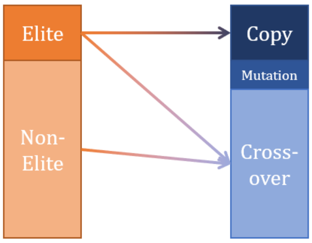


Figure 5.2: The process of creating a new generation in the Genetic Algorithm (GA) from the old population.

the Parameterized Uniform Crossover method is used, as done by Gonçalves in 2005 [27]. The fitness function is defined as the solution that minimizes the volume of the remaining boxes that have yet to be fitted into the van. A schematic overview on how the GA creates a new offspring population is given in Figure 5.2.

The parameters are chosen according to the 2012 paper by Gonçalves [23], as this led to favorable results in their research. The crossover probability is set to 0.8, the mutant probability to 0.1, and the population to approximately 15 times the number of parcels (i.e., $\approx 15 \cdot |P|$). These parameter configuration also resulted in good outcomes for this model, as discussed in Chapter 6.

5.3. Combined model

With both models adjusted, it is now possible to formulate the combined model, that consists of several steps. To provide clarity, this section offers a logical schedule and pseudo-code for the model.

5.3.1. Logical schedule

A logical schedule of the full model can be found in Figure 5.3. In this section, the different steps as presented in this logical schedule will be elaborated on one by one.

Input In order to run the model, certain inputs are required, which consist of several parts:

1. Van properties; The dimensions of the specific parcel delivery van that needs to be packed, along with the number of shelves and their placement.
2. Parcel properties; The dimensions of all the parcels, as well as the delivery order. The input requires a sorted list of parcels, with the first one being the last to be delivered (LIFO).
3. A value for as; This indicates the percentage of the length and width of the parcels that must be covered by a surface below them for stability.
4. Two time limits; One for how long the GA should search before terminating, and one for the optimization time for the objective function of the MILP.

Heuristic The first step of the model is to find a feasible solution using the heuristic solution method. To create a solution that maintains the LIFO order as good as possible, two settings are used: the distance baseline selected in Section 5.2.1 and an input order coherent with the LIFO order, as presented in Section 5.2.2. If this step leads to a feasible packing, the model proceeds to the next step, which involves using the feasible solution as an initial solution for the MILP. However, if no feasible solution is found with these settings, the model will attempt the heuristic again, but with different settings.

Heuristic with changed baseline If the heuristic described above fails to find a solution, a change is made to the distance baseline. As mentioned in Section 4.3.2, using the original DFTRC-2 heuristic can result in a more compact solution with fewer gaps. By measuring the distance from the point (L, W, H) instead of the line $(\frac{L}{2}, W, z), z \in [0, H]$, a more compact solution may be achieved (although it is not guaranteed). If a feasible solution is found using this method, the model moves on to the MILP. If this approach also fails to produce a feasible solution, the model then runs the heuristic with a GA over the input.

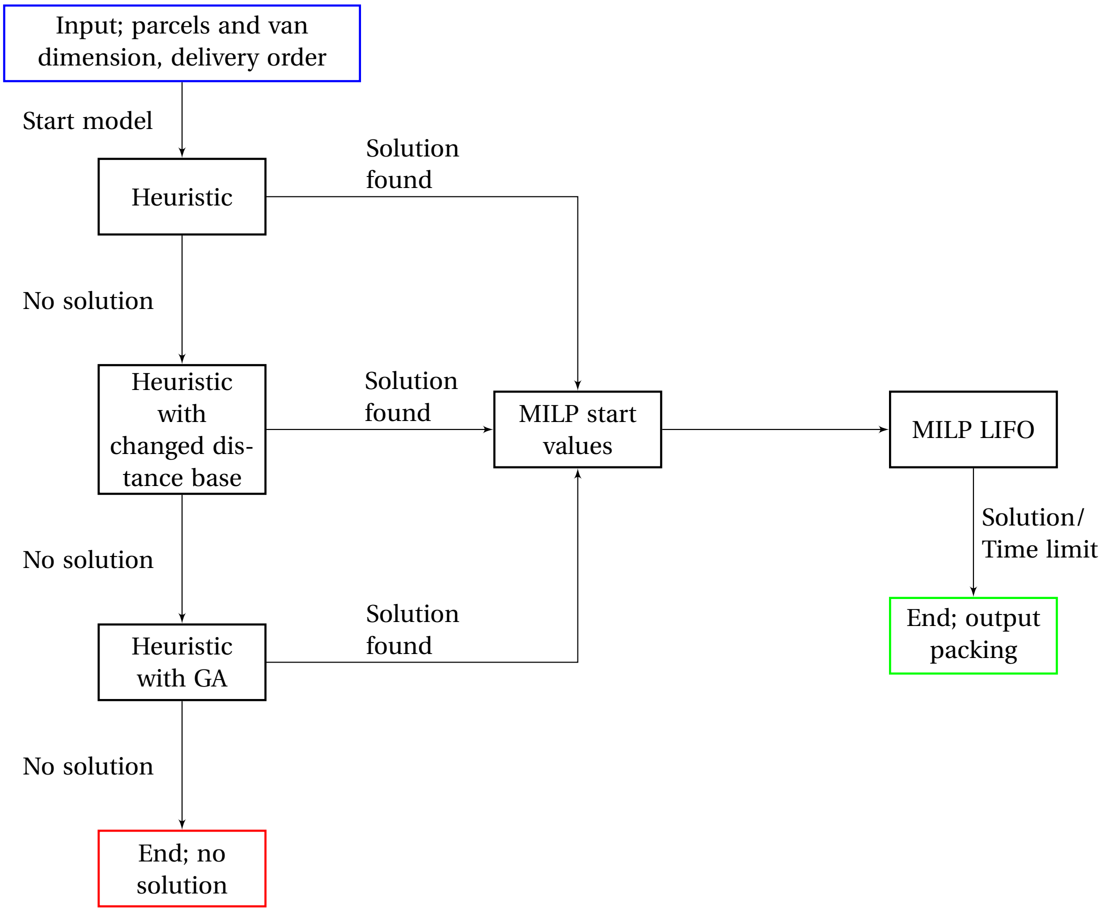


Figure 5.3: Logical schedule of the combined model

Heuristic with GA If the previous two steps failed to produce a feasible solution, the model changes the input order using a GA, as described in Section 5.2.3. The first time limit from the input is considered in this step. If the GA does not succeed to find a feasible solution within this time limit, the model terminates, indicating that no feasible solution has been found for the given input. However, if a feasible solution is found, the model proceeds to the MILP.

End; no solution If the heuristic fails to find a solution, the model terminates. An option is to remove one or more large parcels and run the model again. The parcels that are not included in the model should then be assigned to a different delivery van.

MILP start values After the heuristic finds a solution, the model proceeds to the next step, which involves running the MILP. The MILP uses two data points per parcel obtained from the heuristic; the (x, y, z)-coordinates of the LBBC and the orientation of the parcel to create a complete packing of the van. During this step, the model assigns values to all other variables (such as a_{ik}, X_i, l_{xi}, etc.) based on the packing. Once this is completed, the model begins optimizing the objective functions.

MILP LIFO In this step, the objective is to minimize the number of parcels that cannot be reached if the parcel deliverer arrives at their destination. This is quantified in the MILP through the minimization of the function $\sum_{i \in P} \delta_i$. Details of this objective function can be found in Section 5.1.2. The optimization process stops either when a predetermined time limit is reached, or when an optimal solution is found. Once this step is completed, the model terminates.

End; output packing After completing all the steps, the model terminates. The output is a value for all the variables of the MILP. This can be translated to a value for the objective function and to create a visual representation of the packing by using the coordinates and orientation of the parcels.

5.3.2. Pseudo-code

To finalize the description of the full model, a pseudo-code of the model can be found in Procedure 2, which follows the same steps as those explained previously.

Procedure 2 Combined model for 3L-BPP

Input: Sorted list P with parcels to be packed, dimension L, W and H for the van, shelves dimensions, area support parameter as and time limit A and B.

Output: A Placement containing the (x, y, z)-coordinates for the LBBC and the orientation of every parcel.

```
Heuristic method with $\rho_i = \max\{(\frac{L}{2} - x_i - X_i)^2 + \varepsilon \cdot (W - y_i - Y_i)^2$
if Placement is feasible then
    Placement = Placement found by the heuristic
else
    Heuristic method with $\rho_i = \max\{(\frac{L}{2} - x_i - X_i)^2 + (\frac{W}{2} - y_i - Y_i)^2 + (\frac{H}{2} - z_i - Z_i)^2$
    if Placement is feasible then
        Placement = Placement found by the heuristic
    else
        Heuristic method with GA on the input order and $\rho_i = \max\{(\frac{L}{2} - x_i - X_i)^2 + (\frac{W}{2} - y_i - Y_i)^2 + (\frac{H}{2} - z_i - Z_i)^2$ until time limit A is reached
        if Placement is feasible then
            Placement = Placement found by the heuristic
        else
            return False                    ▷ No solution has been found within the time limit
        end if
    end if
end if
check Placement is a solution to the MILP with soft constraints
minimize the objective $\sum_{i \in P} \delta_i$ in the MILP with Placement as input until optimized or time limit B is reached
return Placement
```

6 Results

This chapter aims to evaluate the model presented in Chapter 5 by conducting tests with real-world data. Specifically, the testing process utilizes historical data of particular rides, which encompasses information such as parcel dimensions, delivery order, and van dimensions. As the model is exclusively tailored to address the 3D-Bin Packing Problem with Loading Constraints (3L-BPP), a problem that is uniquely defined in this thesis, no other models currently exist to provide benchmark results for comparison. Consequently, the evaluation will focus on assessing the performance of the model in terms of both speed and accuracy, rather than comparing it against existing benchmarks.

The chapter begins with a description of the data set characteristics, model parameters, and implementation features. Then, results are presented about the first stage of the model, which uses a heuristic approach. This is followed by a comparison of the heuristic approach with the MILP. Thereafter, the data of the combined model is collected and analyzed. Then, the model is applied to a real-world situation where a van is loaded using the outcome of the model. Finally, a brief preliminary investigation of alternative model options is provided.

The model is implemented in Python 3.10, using Gurobi Optimizer© version 9.5.2 buildv9.5.2rc0 (win64) for the optimization of the MILP. The computer that is used uses a 11th Gen Intel(R) Core(TM) i5-1145G7 @ 2.60GHz with a Windows operating system.

6.1. Data set and parameter settings

In this section, it is defined what the data consists of, how the data is collected and selected, the parameter settings and what preprocessing is done.

Data content The model is tested on data provided by PostNL Holding B.V., a leading postal and parcel delivery company in the Netherlands, that serves approximately 50 to 55% of the consumers parcel delivery service of the country [28]. The company provided four datasets, each containing information on all rides conducted by a specific parcel delivery depot on a given day. These datasets include parcel dimensions and corresponding delivery orders for every ride. Table 6.1 provides an overview of the properties of each dataset, which collectively comprise 739 unique rides.

Data selection To ensure the data is diverse, the set includes a mix of different locations, days of the week, and periods throughout the year. The choice in datasets is made to have a wide spectrum of different inputs during the testing. There are three locations inside the agglomeration of cities in Netherlands (Randstad) and one in the rural area. For the purpose of this research, it was opted to gather more data from depots located in urban areas. This decision was based on the observation that, on average, rides in urban areas tend to contain a greater number of parcels than those in rural areas. As the packing problem becomes increasingly complex as the number of parcels increases, studying the performance of the model in high-density scenarios can provide valuable insights into its capabilities and limitations. Therefore, by focusing on these more challenging

Table 6.1: Overview of the properties of the different data sets used to generate results on the model.

Location	Area Type	Date	Weekday	No. rides	Mean no. parcels	Mean fill rate
Zwolle	Rural	20-5-2022	Friday	205	141	26.2%
Den Hoorn	Urban	15-11-2022	Tuesday	182	155	27.7%
Amsterdam	Urban	10-1-2022	Monday	172	159	28.1%
Sassenheim	Urban	30-11-2022	Wednesday	180	228	38.4%

settings, it is possible to gain a better understanding of the strengths and weaknesses of the model.

Notably, the datasets included two dates in November. This is typically the busiest month for PostNL due to holidays and special occasions such as Black Friday, Cyber Monday and 'Sinterklaas', and already the early shopping for Christmas. Conducting tests on the November datasets, which include a mid-month dataset and an end-of-month dataset, is particularly interesting due to the exceptionally high number of parcels delivered during this period, for the same reason that urban areas are of particular interest.

Parameters settings Appendix A specifies a standard size for the delivery van loading space of $(L \times W \times H) = (192 \times 323 \times 178)$ cm. The area support parameter, as, is taken equal to 0.9 and applied uniformly across all data sets. The vans contain four shelves, each with a length of $\frac{L}{4}$ and a width of W. The shelves are placed on the left and right at heights of $\frac{H}{3}$ and $\frac{2H}{3}$. The fill rate for one ride is defined as $\frac{\sum_i p_i \cdot q_i \cdot r_i}{L \cdot W \cdot H} \cdot 100\%$, where the sum is over all parcels i in that specific ride.

Preprocessing Before running the model, certain data preprocessing steps are required. Firstly, parcels lacking information on their length, width, or height are removed from the data set. Secondly, parcels that exceed the available shelf space are classified as 'large' and placed in the aisle without incurring penalties based on their position in the aisle, as determined by the γ_{ik} variable.

Data merging In the next section, it will be shown that the performance of the various data sets is similar, with no significant differences between them. While rides originating from Sassenheim have a higher fill rate compared to others, rides with similar fill rates across different data sets also generate similar results. This similarity is also observed in the solution method used. Therefore, the full data set can be treated as a single set with varying data for each ride, instead of dividing it into four separate sets.

6.2. Results for the Heuristic Solution Method

In this section, the computational results for the heuristic solution method, as constructed in Chapter 4 and adjusted in Section 5.2, are examined. First, the different solution options are explored, then the results on the runtime and the objective function value are analyzed and concluded.

6.2.1. Solution options

When looking at the heuristic solution method described in Section 5, the first part consists of finding a feasible solution. There are four possible outcomes of this step, which will be numbered and referred to by their respective number for the remainder of this chapter.

1. The heuristic with the LIFO input order and the adjusted distance measure found a solution.
2. The heuristic with the LIFO input order and the original DFTRC-2 distance measure found a solution.
3. The heuristic with a GA incorporated to determine the input order and the original DFTRC-2 distance measure found a solution.
4. No solution was found within the time limit using the same settings as in solution option 3.

Table 6.2: Results after running the heuristic on the data set of 789 rides. The properties are divided into which solution option found a feasible packing.

Solution	Percentage of rides	Av. runtime	Av. objective function value	Av. wrongly placed parcels
1	97.7%	0.4 sec	11	6%
2	1.2%	1.2 sec	165	54%
3	1.1%	38.1 sec	205	66%
4	0%	-	-	-

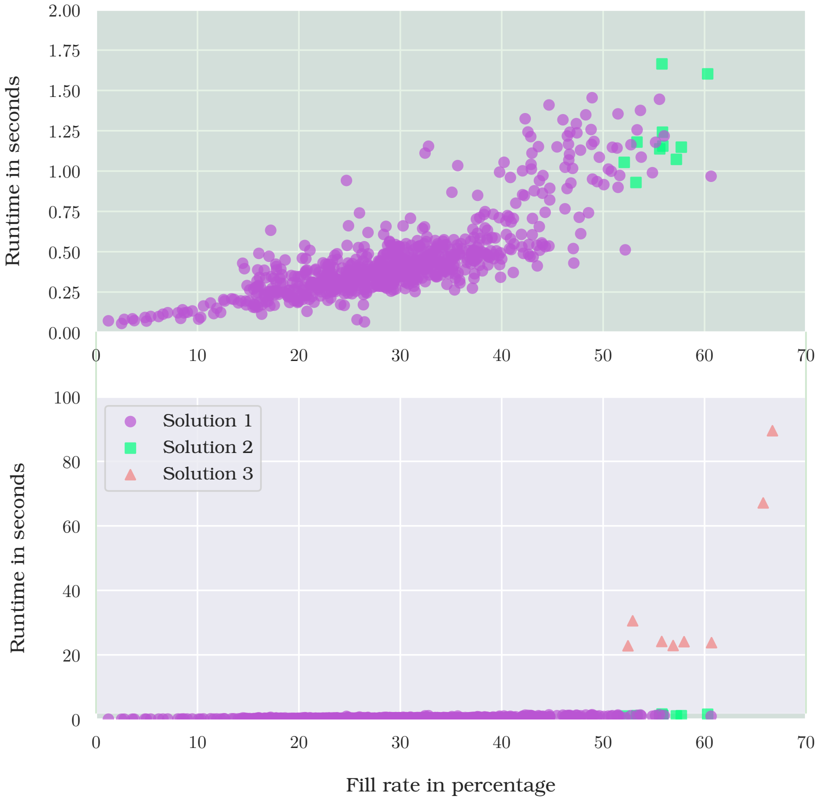


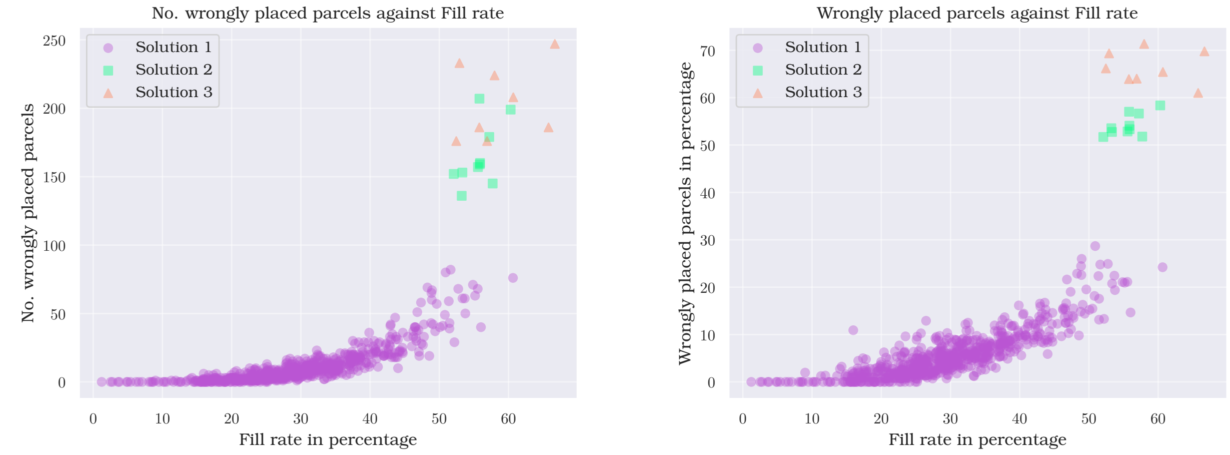
Figure 6.1: Results after running the heuristic on the data set of 789 rides, where every point represents one ride and the solution options are showed with different colors/shapes. The lower plot shows the runtime needed to create a feasible packing in seconds on the y-axis, plotted against the fill rate of the van on the x-axis. The upper plot shows a zoom from this plot for only the solutions up to 2 seconds.

As described, the model will also follow these ordered steps to find a feasible packing. It begins with the first option and, if it is unable to find a feasible solution, proceeds to the next option and continues the process until option 4.

6.2.2. Runtime

The results of the feasible solutions of the heuristic are shown in Table 6.2. The heuristic successfully finds a solution for 721 out of 739 rides (97.7%) using the 1st option. The quickness with which these 1st option solutions are generated is noteworthy, with all solutions of this type being found within just 1.5 seconds, and an average time of only 0.4 seconds.

In Figure 6.1, one sees for every ride in a different color/symbol which solution option ([1/2/3]) was successful to find a feasible solution. Here, the runtime in seconds is plotted against the fill rate in percentage. The lower plot shows the results of all the different rides, going from 0 to 100 seconds. The upper plot shows a zoom of the range, going from 0 to 2 seconds. One can see that the first solution option is adequate for all



(a) On the y-axis the number of wrongly placed parcels within the packing is shown, plotted against the fill rate of the van on the x-axis.

(b) On the y-axis the percentage of wrongly placed parcels within the packing is shown, plotted against the fill rate of the van on the x-axis.

Figure 6.2: Results after running the heuristic on the data set of 789 rides regarding the wrongly placed parcels. Every point represents one ride.

rides up to a fill rate of 50%. However, beyond that point, some rides require the second or third option to find a feasible solution. One can conclude that, as the fill rate gets higher, the model is less likely to find a solution using the 1st option. It is worth noting that the model always managed to find a solution within 90 seconds, so the fourth solution option did not occur for the entire data set.

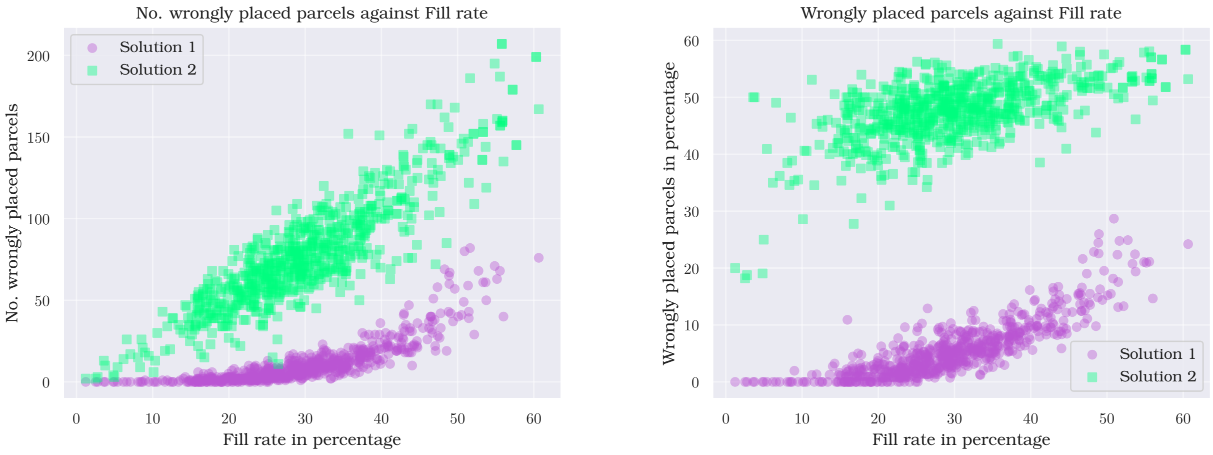
6.2.3. Objective function value

To evaluate the validity of the initial solution of the heuristic, the values of the objective function outlined in Section 5.1.2 are analyzed. The objective function is designed to minimize the value of $\sum_{i \in P} \delta_i$, which represents the number of parcels that are incorrectly loaded into the van according to the LIFO filling principle.

Table 6.2 displays the mean objective function value for each solution type. The results can also be found in Figure 6.2a, where the fill rate is plotted against the number of wrongly placed parcels. It is apparent that the mean value for the first solution type is respectively 15 and 18 times smaller than the second and third solution option. To investigate whether a lower number of parcels for these rides could be the reason behind this result, also the percentage of wrongly placed parcels is included. This is calculated as $\frac{\sum_{i \in P} \delta_i}{|P|} \cdot 100\%$. This is included for the comparison, as extra parcels will naturally lead to extra wrongly placed parcels. This can be found in Figure 6.2b, where the percentage of wrongly placed parcels is plotted against the fill rate of the ride in percentage.

Figure 6.2 reveals that for Solution 1, solutions in the higher segments of the fill rate have a higher objective function value compared to those with a lower fill rate. Nonetheless, the objective function values given by Solution 2 and Solution 3 are significantly higher, even when corrected for the number of parcels inside the van.

It is important to note that the comparison between the two solutions is not entirely fair, as the second solution is only used after the first one fails to find a feasible solution. Therefore, the rides being compared are not identical. To address this, Figure 6.3a shows the results of using both the first and second methods on the same set of rides. Only rides for which both solution method 1 and 2 were able to find feasible solutions are included in this analysis, which corresponds to 709 rides. The x-axis represents the fill rate, while the y-axis represents the number of parcels that are wrongly placed. Overall, the first solution has a significantly lower number of wrongly placed parcels than the second solution, with an average difference of 65. Figure 6.3b also displays the percentage of wrongly placed parcels for both options against the fill rate, which makes it even clearer that the first option performs significantly better than the second. The average percentage difference between the number of wrongly placed parcels per van is 41.3%.



(a) On the y-axis the number of wrongly placed parcels within the packing is shown, plotted against the fill rate of the van on the x-axis.

(b) On the y-axis the percentage of wrongly placed parcels within the packing is shown, plotted against the fill rate of the van on the x-axis.

Figure 6.3: Results after running the heuristic on the data set of 789 rides regarding the wrongly placed parcels twice. Once for solution method 1 and once for solution method 2. Every point represents one ride. The 8 rides with solution method 3 are left out of the plot.

Comparing the same rides with the third option is not possible since its initial value is based on the input of solution 2. If solution 2 finds a solution, the model terminates without executing any GA steps. As a result, it is not possible to make a fair comparison between the third option and the other two solutions.

6.3. Results for the Mixed Integer Linear Program

In this section, the results for the MILP, as constructed in Chapter 3 and adjusted in Section 5.1, will be discussed. The findings of the attempts to obtain a feasible solution without employing the heuristic approach are presented.

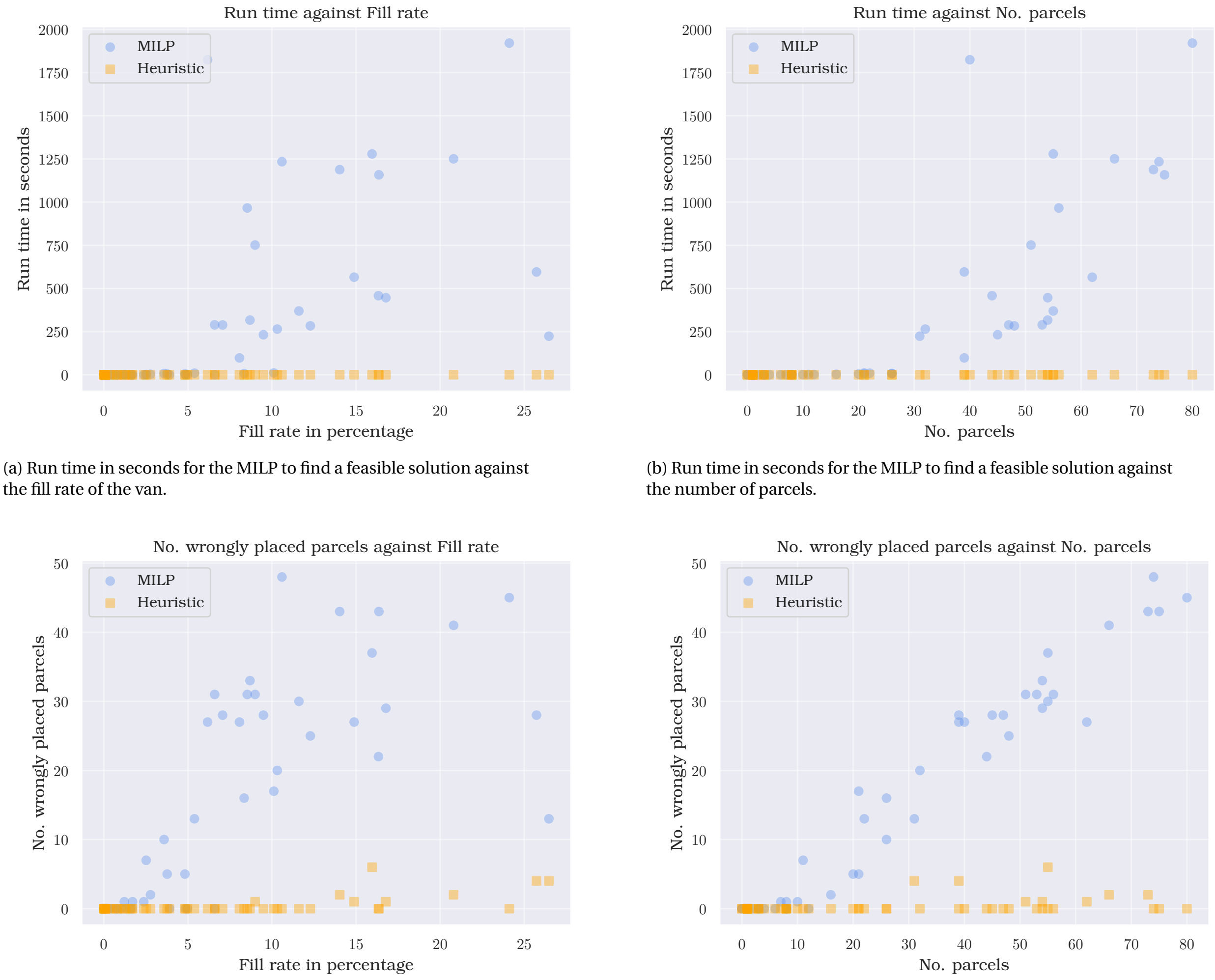
6.3.1. Runtime

As discussed in the previous section, the heuristic algorithm is capable of finding a feasible solution for the rides within a matter of seconds. This raises the question of whether this step is necessary or if the MILP can achieve similar results. Therefore, the results are collected for finding a feasible solution by the MILP.

In Figures 6.4a and 6.4b the runtime of the MILP in seconds is plotted against both the fill rate and the number of parcels of the rides. There are, in contrast to the heuristic, 60 rides analyzed. The reason for analyzing fewer rides is due to the extensive computation time required for the model.

As illustrated in Figure 6.4b, it is apparent that the runtime of the MILP solution increases significantly, taking up to 1000-2000 seconds (15-30 minutes), once more than 60 parcels are inside the delivery van. This difference is substantial compared to the heuristic solution method, which proves to be quicker and more consistent in runtime. When attempting to find a solution for four rides with roughly 80 parcels, the MILP was unable to find a solution within the time limit of 10,000 seconds. In conclusion, the heuristic approach is a more efficient and consistent method to find feasible solutions compared to the MILP when looking at the runtime.

Figure 6.4a does not reveal a clear pattern. Figure 6.4b, does more clearly show that in most cases, as the number of parcels increases, the runtime also increases. This observation is interesting but also has a potential explanation. The variables and constraints in the MILP are all dependent on the set of parcels P, which causes the runtime of the model to potentially increase exponentially with an increase in the size of P. This is not directly linked to the fill rate, which could suggest that it is independent of it.



(a) Run time in seconds for the MILP to find a feasible solution against the fill rate of the van.

(b) Run time in seconds for the MILP to find a feasible solution against the number of parcels.

(c) Number of wrongly placed parcels in the solution generated by the MILP, plotted against the fill rate of the van.

(d) Number of wrongly placed parcels in the solution generated by the MILP, plotted against the number of parcels.

Figure 6.4: Results after running the MILP to find a feasible solution on the data set of 60 rides. Every point represents one ride.

6.3.2. Objective function value

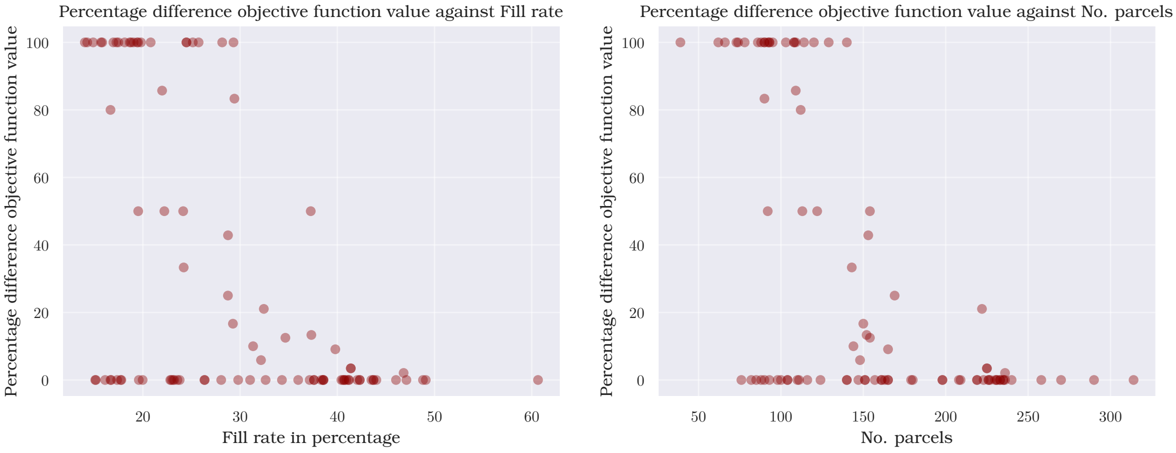
The next step is to evaluate the effectiveness of the initial solutions based on the objective function value. Figure 6.4c the number of wrongly placed parcels for both the MILP and the heuristic method against the fill rate, while Figure 6.4d shows the same comparison against the number of parcels. As depicted in the figures, the heuristic approach outperforms the initial solution of the MILP in terms of maintaining the LIFO order. The value of the objective function increases quickly as the number of parcels increases. Please note that although a feasible solution has been found, the objective function value has not yet been optimized. This can lead to high values, which may not provide much insight.

6.4. Results for the combined model

This section presents the results of the combined model discussed in Section 5.3. Firstly, the check time for the heuristic solution of the MILP is examined. Subsequently, the results of optimizing the solution of the heuristic method by using the MILP are discussed.

6.4.1. Checking feasibility

Once the heuristic has found a solution, the next step is to use this solution as the initial input for the MILP. This entails transforming the information gathered from the heuristic, which is the placement coordinates and orientation of each parcel, into a suitable solution for the MILP. This process involves creating all the



(a) Percentage change plotted against the fill rate.

(b) Percentage change plotted against the number of parcels.

Figure 6.5: Percentage change of the number of wrongly placed parcels from the initial to the optimized solution by the MILP with a maximum runtime of 2000 seconds, using the heuristic solution as a feasible initial solution. Every point represents the percentage change of one ride, for a data set containing 142 rides.

necessary variables, assigning appropriate values to them, and verifying that the constraints are satisfied based on these values. The results are that the runtime increases almost quadratically with the number of parcels, while its correlation with the fill rate is less significant. This is in line with the findings of the previous sections. When the number of parcels increases, the check time increases, with an average time of 0.5 second per parcel and a maximum total time of 500 seconds.

6.4.2. Improving the objective function value

Once the initial solution given by the heuristic has been transformed to a feasible solution for the MILP, the subsequent step, as per the model outlined in Section 3.5, is to minimize the objective function value using the Gurobi Optimizer. Due to the long running time during the collection of these results, only 142 rides are analyzed in stead of the total dataset.

Different time limits of 500, 1000 and 2000 seconds are used to obtain the results. This maximum is decided upon considering that the process of PostNL, the company providing the data, is structured in a manner that allows for ± 1 hour between the availability of data and the actual packing of the parcel delivery van. As a result, because the first steps of the model costs some time as well, a maximum of around 30-minute optimization window is considered feasible before the results need to be collected.

Optimization results after 2000 seconds First, the results after 2000 seconds of optimization are analyzed. Figure 6.5 presents the percentage change in the number of wrongly placed parcels on the y-axis. For 6.5a, the fill rate is plotted on the x-axis while for 6.5b, this is the number of parcels. Note that the rides with optimal values achieved by the heuristic (i.e., an objective function value of 0) are not included in this analysis since the MILP is incapable of improving upon these outcomes. A total of 50 rides from the test data fall into this category.

For some rides with a lower number of parcels, the heuristic incurs minor penalties due to suboptimal parcel placement, which can be addressed by the MILP through parcel replacements, resulting in a 100% improvement. However, as the number of parcels increases, the model requires more time to optimize, which results in a decrease in the percentage change. Beyond 250 parcels, the MILP is unable to enhance the objective value further. This is most likely because replacing even a single parcel would necessitate modifying a considerable number of variables, resulting in extensive computation time that exceeds the available time of 2000 seconds.

Table 6.3: Results on the improvements of the number of wrongly placed parcels using the MILP optimization over time.

Max. runtime MILP	Av. improved no. wrongly placed parcels per ride	Av. improvement objective function value per ride
500 sec	0.86	-29%
1000 sec	1.28	-35%
2000 sec	1.66	-37%

Optimization over time This analysis examines the optimization results achieved under different time limits (500, 1000, and 2000 seconds). Table 6.3 presents two main findings. Firstly, the average improvement in the number of placed parcels per ride was computed by summing the differences between the heuristic objective function value and the value after 500, 1000 and 2000 seconds of optimization, divided by the number of rides. The results suggest an average improvement of only about one parcel per ride. Within the first 500 seconds, the average improvement was 0.8 parcels per ride, with an additional 0.8 parcels per ride over the following 1500 seconds.

Secondly, the average improvement in the objective function value per ride, expressed as a percentage, provides a better understanding of the optimization process. The results show that 29% of the improvement was achieved within the first 500 seconds of optimization, with an additional 6 percentage points achieved after the next 500 seconds. However, from 1000 to 2000 seconds, only 2 extra percentage points were achieved. These results suggest that while additional optimization time can add value, the marginal return on investment diminishes rapidly as the total time increases.

Thirdly, Figure 6.6 provides additional insights into the optimization process over time. The figure shows the objective function value on the y-axis and the number of parcels on the x-axis. The results are plotted for solutions obtained using the heuristic method and the MILP after a maximum runtime of 500 and 1000 seconds. The results with a maximum runtime of 2000 seconds are not included in the figure, as they only exhibit small differences compared to the solutions obtained after 1000 seconds. Two zoomed-in plots of the lower figure are shown at the top of the graph, where $x \in [75, 125], y \in [0, 20]$ and $x \in [135, 185], y \in [0, 20]$ to highlight specific values.

It is interesting to note that the improvement is mostly achieved within the first 500 seconds in the zoomed-in plot for $x \in [75, 125]$. However, in the higher segment, $x \in [135, 185]$, the 500 seconds of optimization only provides an in-between value. Moreover, beyond 200 parcels, the objective function value almost does not decrease. This observation was already established with the results presented in Figure 6.5b.

6.4.3. Try-out of the combined model

An additional real life try-out is conducted to evaluate the performance of the full model, which involved using the heuristic method to find a feasible solution and optimizing the MILP for 2000 seconds. The test is organized at a parcel delivery depot, and a report of this test and the results is provided in Appendix B. The key takeaway from this test is that it successfully demonstrated the feasibility of implementing this model in real-life scenarios.

6.5. Changes to the interior of the van

In addition to providing a feasible packing for parcel delivery vans, the model described in Chapter 5 can also be used to analyze the interior of the van under different conditions. By examining the effects of various factors, such as enlarging the van or adding/removing shelves, this model can help explore new possibilities for improving delivery efficiency. To demonstrate this, five additional scenarios are examined in addition to the one discussed in Section 6.1.

In the first scenario, the dimensions of the van are modified to $(L \times W \times H) = (172 \times 252 \times 141)$ cm, which corresponds to the dimensions of a distinct type of delivery van [29]. This represents a 55% reduction in the volume of the van. In another scenario, the van's shape is modified to be higher than its length to investigate

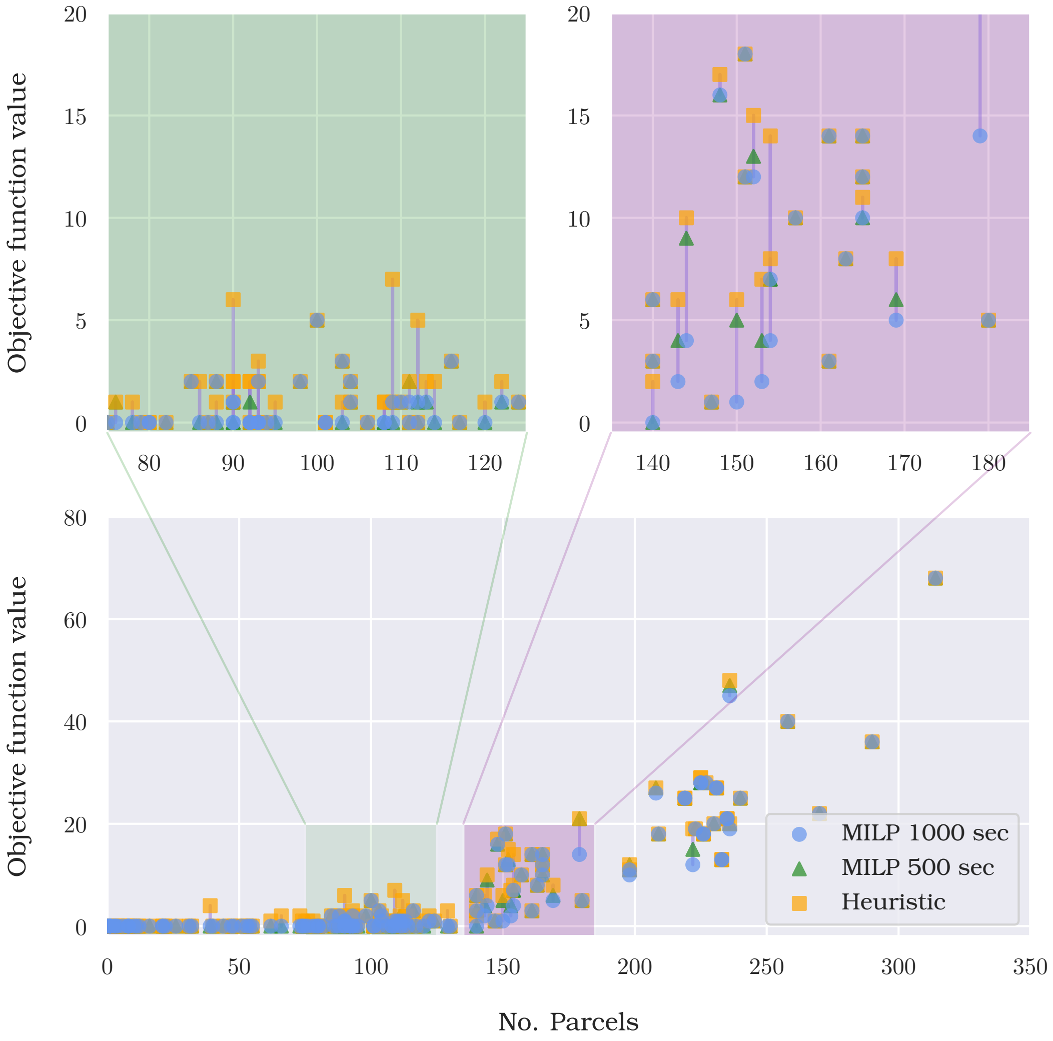


Figure 6.6: Results after running the heuristic and the MILP on a data set of 142 rides regarding the wrongly placed parcels. The initial solution by the heuristic and the optimization after 500 seconds and 1000 seconds are given. The lower plot shows the whole set, the left upper plot a zoom for number of parcel going from 75 to 125, the right upper plot for number of parcels going from 135 to 185.

Table 6.4: Results after running the heuristic method on the data set of 739 rides with different input on the interior of the van. The interior has either more or less shelves then the original (which has 4), or decreased dimensions.

Van interior	Av. objective function value	Percentage of rides no solution found	Average runtime heuristic
Original	15.2	0%	1.0 sec
Dim. 172 × 252 × 141	69.7	14.9%	14.6 sec
Dim. 180 × 257 × 195	31.5	2.4%	41.5 sec
Two shelves	19.9	0%	2.3 sec
Six shelves	11.5	0%	1.2 sec
No shelves	33.7	0.4%	14.6 sec

potential alternative results. The dimensions for this scenario are chosen as $(L \times W \times H) = (180 \times 257 \times 195)$, representing an 18% reduction in size compared to the original. The remaining three configurations retain the original size of the van, but with different shelving arrangements. One setup has two shelves placed at a height of $z = \frac{H}{2}$, while another has six shelves arranged at different heights including two at $z = \frac{H}{4}$, two at $z = \frac{H}{2}$, and two at $z = \frac{3 \cdot H}{4}$. The third configuration, however, has no shelves. Table 6.4 displays the results for the different inputs. Only the initial solution generated by the heuristic is taken into account, so the MILP did not perform any optimization.

Significantly, in scenarios with reduced dimensions, the average value of the objective function increases by 61% despite only an 18% reduction in space, and by 458% with a 55% reduction in space. This increase in objective function value is considerably higher than the reduction in space. It is worth noting that if no solution is found, the objective function value is taken as $|P|$, which represents the number of incorrectly placed parcels.

Another interesting finding is that having the same dimensions but a different number of shelves has a significant influence on the objective function. By going from 4 to 6 shelves in the vans, the average objective function value generated by the heuristic decreases by 25%. While this is only a preliminary investigation, it demonstrates the model's potential to generate useful insights into the optimal interior design for a parcel delivery van.

6.5.1. Other applications

The model has been designed in a general manner, allowing the possibility to be applied beyond the context of parcel delivery vans. It can be applied to packing and unloading tasks in various ways. The only requirement is that there are dimensions for the objects that need to be packed and the object in which they can be packed. If applicable, the loading sequence can also be included, although the model can generate a feasible solution even without this information. This can make the model suited for various applications, including but not limited to warehouses, shop shelves, parcel lockers, and parcel delivery cargo bikes.

7

Conclusion and discussion

The purpose of this thesis was to find an answer to the following research question:

> **Research question**
>
> What modifications can be made to existing parcel packing optimization models to a create feasible packing solution within delivery vans, given a specific set of parcels and delivery sequence?

To tackle this question, a series of steps were taken. First, the problem was defined by using the framework of an existing problem, namely the 3D-Bin Packing Problem (3D-BPP). The criteria used in this problem were modified to focus on the specific problem properties for a parcel delivery van. This led to the definition of seven conditions, which together formed the 3D-Bin Packing Problem with Loading Constraints (3L-BPP).

To construct an analytical model for the 3L-BPP, an existing Mixed Integer Linear Program (MILP) for the 3D-BPP was used as a starting point. The model was adjusted by systematically addressing each of the conditions that needed to be met, and by developing constraints to handle them. Changes were made to allow for the inclusion of shelves, ensure that parcels could not float, guarantee that parcels had adequate bottom support, and maintain the Last-In-First-Out (LIFO) principle. To ensure correctness, a mathematical proof was conducted.

The results showed that the MILP was indeed able to find a feasible solution for the 3L-BPP. As expected for an analytical model solving a problem that is NP-hard, it took a vast amount of time to find this solution as the input increased. For a small number of parcels, less then 30, it was able to find the solution within 5 minutes.As the number of parcels increased, so did the time required to find a solution, resulting in the time limit being reached before a solution could be found for instances with approximately 80 parcels.

To ensure a solution for instances with a higher number of parcels, a novel heuristic method was developed. The Distance to the Front-Top-Right Corner (DFTRC)-2 provided the foundation for this heuristic, as it produces solutions for the 3D-BPP. The model was modified to satisfy most of the constraints of the 3L-BPP, namely to ensure that the parcels do not float and that shelves are included in the solution. This was achieved by ensuring that the Empty Spaces (ES) used to place the parcels were created in a correct manner. Finally, the distance measurement was altered to improve the maintenance of the LIFO order. After implementing this new new measurement method, the number of misplaced parcels decreased significantly. On average, there was a 41.3% reduction in the number of incorrectly placed parcels per ride compared to previous solutions.

The newly developed heuristic successfully found a solution for 97.7% of the rides in the dataset within 3 seconds. To address the remaining 2.3% of the rides, some modifications were made. By using the original distance measurement, 1.2 percentage points of the remaining rides were solved within 2 seconds. For the remaining unsolved rides, an Genetic Algorithm (GA) was incorporated to reorder the input in a structured manner, which led to a feasible solution for all rides within 100 seconds.

In conclusion, the heuristic method significantly reduced the time required to obtain results, minimized the objective function value, and improved the feasibility of finding a solution within the given time limit. By adding the GA, the heuristic was able to ensure a feasible solution for all rides.

Ultimately, the proposed problem was solved by combining the heuristic and the MILP into a single model. The heuristic was first used to create a feasible solution, which was then optimized using the MILP. For rides with a small number of parcels (up to about 125), this approach resulted in solutions with minimal incorrectly placed parcels. In the cases where the heuristic produced a solution with some misplaced parcels, the MILP optimization process was able to significantly improve the solution within a short time frame.

Even when the number of parcels increased to approximately 200, the MILP still managed to produce improved solutions, although at the cost of longer run times. For the highest number of parcels (above 200), the MILP optimization process generally did not improve the solution given by the heuristic method. This result is not surprising, given that this problem is classified as NP-hard. Overall, the findings show that optimization by MILP can effectively reduce the number of wrongly placed parcels, but the benefits are less pronounced as the number of parcels increases.

A try-out was preformed to assess the feasibility of implementing the model in a real-world scenario. This gave the fruitful result that there surely is a possibility to apply the model at parcel delivery companies if a few adjustment are made to the model.

Finally, although the primary objective of the model was to find feasible packing solutions for parcel delivery vans, it was discovered that the model could have other potential applications as well. The model has the capability to suggest optimal ways to design the parcel delivery vans to make the most out of the available space. Some preliminary tests showed that if PostNL B.V. decided to switch from 4 to 6 shelves inside their vans, the average percentage of wrongly placed parcels per ride, as found by the heuristic, would be reduced by 25%. This application is fairly interesting and could be further investigated. Next to that, the models flexible input makes that it can be used to pack other types of objects as well, such as the shelves in a warehouse or a store.

This thesis presented a novel problem description and no other existing solution methods can be used for comparison. To better assess the performance of the model, other solution methods should be created and tested against a similar data set. Therefore, it cannot be concluded that the final model is the optimal strategy for creating a feasible packing in general. However, it can be stated that among the solution methods explored in this thesis, using the combined model as presented in Chapter 5 gives the optimal strategy to create a feasible packing.

The results clearly indicate that the heuristic is notably faster in finding solutions than the MILP. Therefore, when considering possible enhancements for this model, it would be beneficial to explore ways in which the heuristic can generate a better initial solution. It may be acceptable to sacrifice some time for this purpose since, for most applications of this model, an immediate solution is not necessarily required. One approach to improving the model would be to generate an ES only if the placement conforms to the LIFO principle. Additionally, the model currently only considers square shapes, and expanding it to include other shapes could be a potential future direction for the development of the model.

To sum up, this thesis presented a model to create a feasible packing inside a delivery van, given a set of parcels and a complementary delivery sequence. Tests have shown that using a heuristic method to generate an initial solution, which was further optimized by a MILP, led to favorable results. All in all, the model performed well on a large set of test data, has the potential to be applied in real-life scenarios, and can be extended to other applications. These results create a foundation for further research in this field.

8

Bibliography

[1] S. Chevalier, “Retail e-commerce sales worldwide from 2014 to 2026,” July 2022. Statista.com.

[2] Effigy Consulting, “Courier, express and parcel (cep) market volume in netherlands from 2014 to 2021,” 2022. statista.com.

[3] PostNL Holding B.V. , “Research done within postnl,” 2021.

[4] P. K. Ariningsih, T. Iswari, K. D. Poetra, and Y. M. K. Aritonang, “Sequential routing-loading algorithm for optimizing one-door container closed-loop logistics operations,” *Jurnal Optimasi Sistem Industri*, vol. 19, pp. 122–132, 2020.

[5] S. Martello, D. Pisinger, and D. Vigo, “Three-dimensional bin packing problem,” *Operations Research*, vol. 48, pp. 256–267, 2000.

[6] A. R. Brown, *Optimum packing and depletion: the computer in space - and resource-usage problems*. American Elsevier, 1971.

[7] H. Dyckhoff, “A typology of cutting and packing problems,” *European Journal of Operational Research*, vol. 44, pp. 145–159, 1990.

[8] 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.

[9] L. V. Kantorovich, “Mathematical methods of organizing and planning production,” 1939.

[10] L. Faina, “A survey on the cutting and packing problems,” *Bolletino dell Unione Matematica Italiana*, vol. 13, pp. 567–572, 2020.

[11] J. A. George and D. F. Robinson, “A heuristic for packing boxes into a container,” *Computers and Operations Research*, vol. 7, pp. 147–156, 1980.

[12] W. F. Maarouf, “A new heuristic algorithm for the 3d bin packing problem,” *Innovations and Advanced Techniques in Systems, Computing Sciences and Software Engineering*, 2008.

[13] E. Bischoff and G. Wäscher, “Cutting and packing,” *European Journal of Operational Research*, vol. 84, pp. 503–505, 1995.

[14] F. Massen, Y. Deville, and P. V. Hentenryck, “Pheromone-based heuristic column generation for vehicle routing problems with black box feasibility,” vol. 7298 LNCS, pp. 260–274, 2012.

[15] Z. Chen, M. Yang, Y. Guo, Y. Liang, Y. Ding, and L. Wang, “The split delivery vehicle routing problem with three-dimensional loading and time windows constraints,” *Sustainability (Switzerland)*, vol. 12, 2020.

[16] M. Gendreau, M. Iori, G. Laporte, and S. Martello, “A tabu search algorithm for a routing and container loading problem,” *Transportation Science*, vol. 40, pp. 342–350, 2006.

[17] V. Lurkin and M. Schyns, “The airline container loading problem with pickup and delivery,” 2015.

[18] S. Jakobs, “On genetic algorithms for the packing of polygons,” *European Journal of Operational Research*, vol. 88, pp. 165–181, 1996.

[19] E. Hopper, “Two-dimensional packing utilising evolutionary algorithms and other meta-heuristic methods,” 2000.

[20] K. Karabulut and M. I. Murat, “A hybrid genetic algorithm for packing in 3d with deepest bottom left with fill method,” *Lecture Notes in Computer Science*, 2004.

[21] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*. Springer, third edition ed., 2008.

[22] S. Pedruzzi, L. P. A. Nunes, R. D. A. Rosa, and B. P. Arpini, “A mathematical model to optimize the volumetric capacity of trucks utilized in the transport of food products,” *Gestao e Producao*, vol. 23, pp. 350–364, 2016.

[23] J. F. Gonçalves and M. G. Resende, “A biased random key genetic algorithm for 2d and 3d bin packing problems,” *International Journal of Production Economics*, vol. 145, pp. 500–510, 2013.

[24] K. Deb, “An introduction to genetic algorithms,” *Sādhanā*, vol. 24, pp. 293–315, 1999.

[25] M. Gajda, A. Trivella, R. Mansini, and D. Pisinger, “An optimization approach for a complex real-life container loading problem,” *Omega (United Kingdom)*, vol. 107, 2022.

[26] M. N. Zeilinger, M. Morari, and C. N. Jones, “Soft constrained model predictive control with robust stability guarantees,” *IEEE TRANSACTIONS ON AUTOMATIC CONTROL*, p. 1, 2014.

[27] J. F. Gonçalves, J. J. de Magalhães Mendes, and M. G. Resende, “A hybrid genetic algorithm for the job shop scheduling problem,” *European Journal of Operational Research*, vol. 167, no. 1, pp. 77–95, 2005.

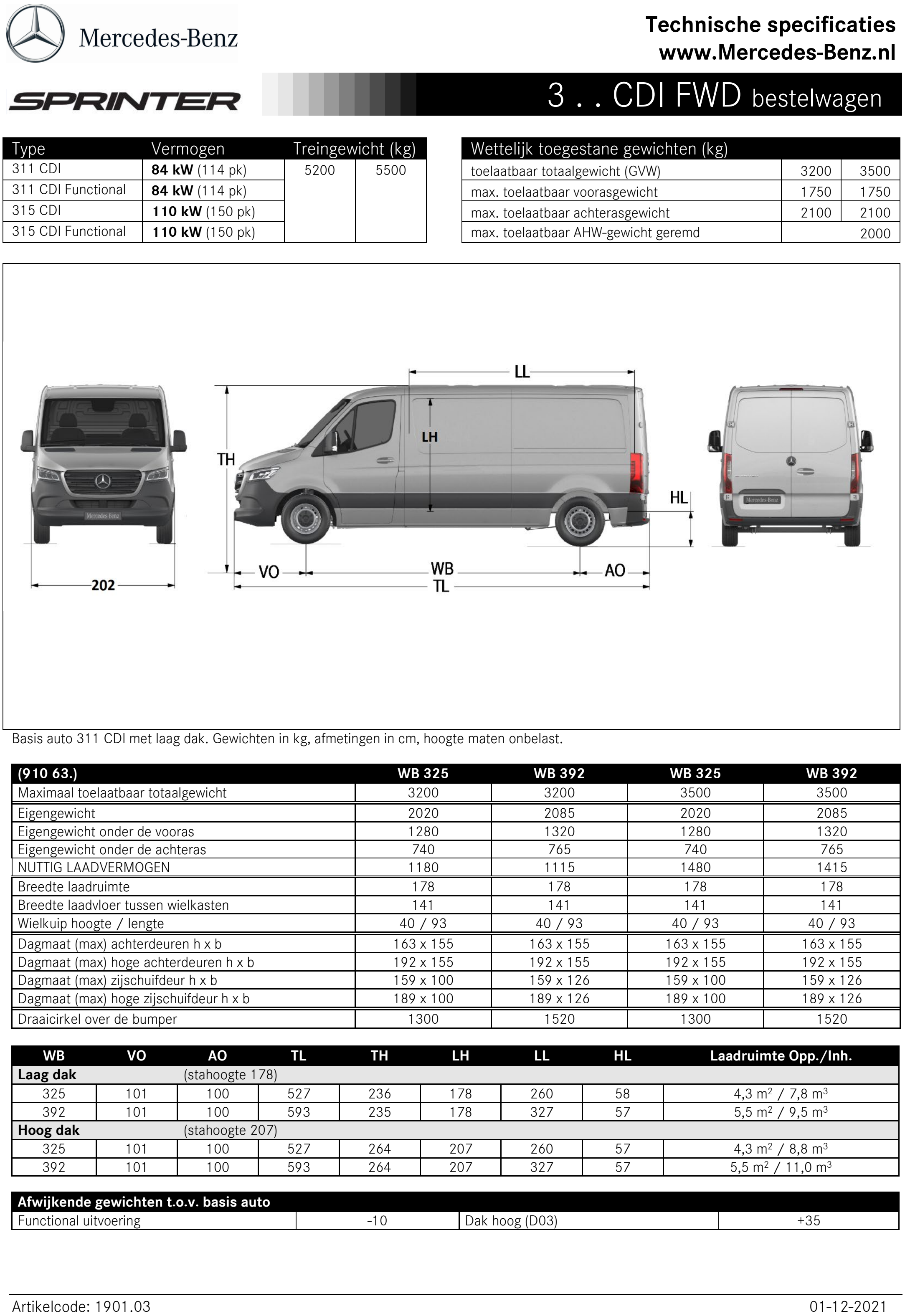
[28] Autoriteit Consument en Markt, “Post- en pakketmonitor 2021,” 2021.

[29] london-man-van, “Medium van dimensions,” May 2023. London-man-van.com.

Appendices

A

Technical specifications delivery van



Mercedes-Benz

Technische specificaties
www.Mercedes-Benz.nl

SPRINTER

3 . . CDI FWD bestelwagen

Type	Vermogen	Treingewicht (kg)	
311 CDI	**84 kW** (114 pk)	5200	5500
311 CDI Functional	**84 kW** (114 pk)		
315 CDI	**110 kW** (150 pk)		
315 CDI Functional	**110 kW** (150 pk)		

Wettelijk toegestane gewichten (kg)		
toelaatbaar totaalgewicht (GVW)	3200	3500
max. toelaatbaar voorasgewicht	1750	1750
max. toelaatbaar achterasgewicht	2100	2100
max. toelaatbaar AHW-gewicht geremd		2000

Basis auto 311 CDI met laag dak. Gewichten in kg, afmetingen in cm, hoogte maten onbelast.

(910 63.)	WB 325	WB 392	WB 325	WB 392
Maximaal toelaatbaar totaalgewicht	3200	3200	3500	3500
Eigengewicht	2020	2085	2020	2085
Eigengewicht onder de vooras	1280	1320	1280	1320
Eigengewicht onder de achteras	740	765	740	765
NUTTIG LAADVERMOGEN	1180	1115	1480	1415
Breedte laadruimte	178	178	178	178
Breedte laadvloer tussen wielkasten	141	141	141	141
Wielkuip hoogte / lengte	40 / 93	40 / 93	40 / 93	40 / 93
Dagmaat (max) achterdeuren h x b	163 x 155	163 x 155	163 x 155	163 x 155
Dagmaat (max) hoge achterdeuren h x b	192 x 155	192 x 155	192 x 155	192 x 155
Dagmaat (max) zijschuifdeur h x b	159 x 100	159 x 126	159 x 100	159 x 126
Dagmaat (max) hoge zijschuifdeur h x b	189 x 100	189 x 126	189 x 100	189 x 126
Draaicirkel over de bumper	1300	1520	1300	1520

WB	VO	AO	TL	TH	LH	LL	HL	Laadruimte Opp./Inh.
Laag dak		(stahoogte 178)						
325	101	100	527	236	178	260	58	4,3 m^2 / 7,8 m^3
392	101	100	593	235	178	327	57	5,5 m^2 / 9,5 m^3
Hoog dak		(stahoogte 207)						
325	101	100	527	264	207	260	57	4,3 m^2 / 8,8 m^3
392	101	100	593	264	207	327	57	5,5 m^2 / 11,0 m^3

Afwijkende gewichten t.o.v. basis auto			
Functional uitvoering	-10	Dak hoog (D03)	+35

Artikelcode: 1901.03 01-12-2021

B Try-out of the model

To evaluate the performance of the model in a real-world scenario, a test was conducted at one of the parcel delivery depots of PostNL B.V., the same company that provided the data for the results in Chapter 6. The test ride had 186 parcels, with a fill rate of 33.7%. Using the first solution method, the model found a feasible packing within 0.36 seconds. The initial solution was improved by the MILP, going from $\sum_{i \in P} \delta_i = 10$ to $\sum_{i \in P} \delta_i = 9$, within a time limit of 2000.

In order to translate the solution of the model into a practical solution, the van was divided into multiple sections, each designated by a number and a letter. These numbers corresponded to specific areas within the van as follows:

0. the space in front of the side door
1. The space beneath the right shelf
2. The space beneath the left shelf
3. The area on the right lower shelf
4. The area on the left lower shelf
5. The area on the right upper shelf
6. The area on the left upper shelf
7. The aisle

To indicate the appropriate placement of parcels within the van, the letters F, M, and B were used, representing Front, Middle, and Back, respectively. For instance, the combination M6 would indicate that the parcel should be placed in the middle of the van, on top of the upper left shelf. The number 0 is the only compartment that does not require a letter, as it already constitutes a single compartment. A visualization of how one of the shelves of the van is compartmentalized can be seen in Figure B.1.

The barcode of each parcel was linked to the specific letter-number combination, and this information was loaded onto a digital glove. The glove featured a scanner and a screen, allowing the user to scan a parcel and instantly display its corresponding placement code on the screen. Visuals of this process can be found in Figure B.2. By using this technique, the process of loading the van proceeded smoothly. All of the compartments had enough available space. This successful test demonstrated the feasibility of implementing this model in real-life scenarios.

Recommended for parcel delivery companies is that with a few adjustment the model can be made more applicable in real life, such as marking items as fragile, grouping parcels from the same household, letting soft parcel not stand underneath other parcels, etc. These relative small adjustments are outside the scope of this research, but could be very useful in real life applications.

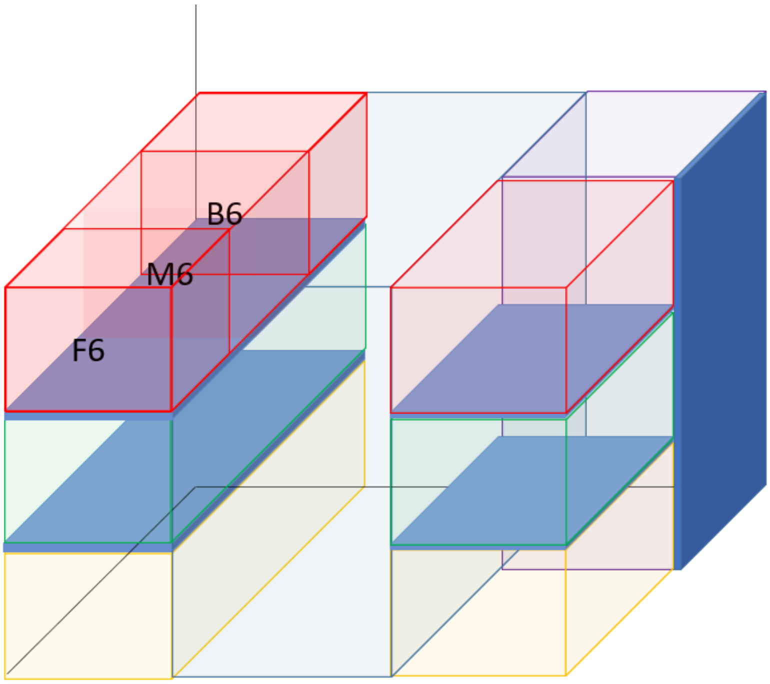
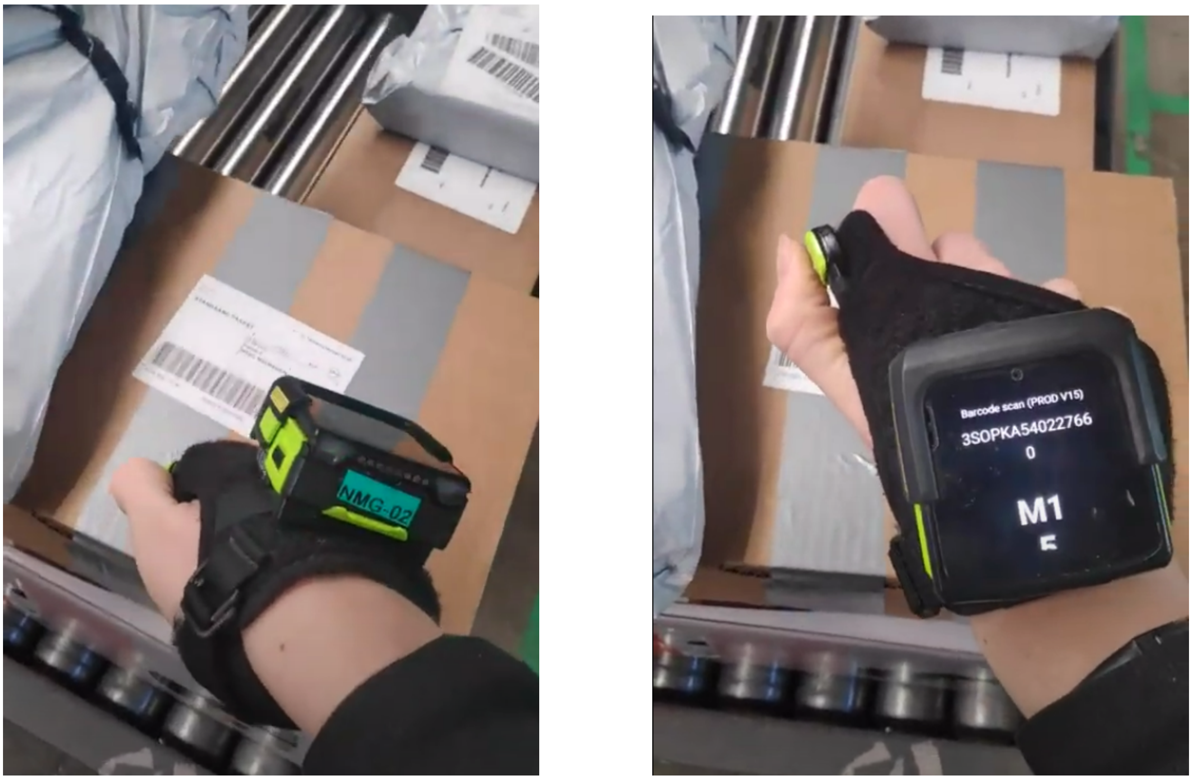


Figure B.1: Compartmentalization of one shelf of the van.



(a) Step one in the process: scanning the parcel.

(b) Step two in the process: receiving the placement information.

Figure B.2: Two steps when using the model in a real life packing situation.