Three-Dimensional Bin Packing Method for Intelligent Sorting of Over-Sized Parts in a Robot Cell

by

Bilal Azam

A thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Department of Mechanical Engineering

University of Manitoba

Winnipeg, Canada

Abstract

This thesis presents the development of a model for a unique bin packing and stacking of parts. The methodology developed is applied to a problem in a manufacturing industry wherein roof trusses for residential and industrial building are assembled after being cut on a computer-controlled saw. The truss members will be stacked into several bins using a custom developed algorithm that optimizes several objectives, namely: bin packing, stability of sorted and stacked parts within the bin, and sequencing of stacking within the bins to facilitate assembly process. The testing, verification and performance evaluation using simulation/implementation in a robotic cell is also presented. The goal is to optimize bin packing by allowing truss members to overhang beyond the edges of the bins while maintaining the stack stability. The overhanging of parts is never considered in standard bin packing problem. The application considered in this thesis will benefit from including this option. The method is simulated in 'Unity', a gaming software and 'Octopuz', a simulation software. The methodology has been implemented in a robot cell. The motivation for this work as noted earlier stems from a need in a local truss manufacturing industry. The methodology is capable of providing a complete automation solution for sorting truss members in similarly operating lumber industry. Based on the tests, the methodology has shown very promising results with 30% decrease in stacking time per part as compared to a previously completed work [1].

Acknowledgements

First, I would like to thank my supervisor and mentor, Dr. Subramaniam Balakrishnan for his guidance and support throughout my studies and research. His direction and care has always been a source of motivation for me.

I would also like to pay gratitude to University of Manitoba for its financial support to compete my master's degree.

I would also like to thank:

- Other thesis committee members, Dr. Qingjin Peng and Dr. Shaahin Filizadeh
- David Tataryn for his work and assistance for successful completion of the project
- Matthew Driedger for his help and support during the research
- To all the friends for adventures, late nights and beautiful memories that we have shared.
- My parents for their support, motivation and love during my master's degree and throughout my life.

Dedication

To my parents and siblings for their encouragement and love throughout my life.

Contents

List of Tables

List of Figures

1. Introduction

Automation in manufacturing industry has shown tremendous growth over the past few years as a result of advancement in the field of automation. Higher product quality is greatly associated with automation while other major advantages include: increased productivity with quality output, enhanced consistency of products or processes, and reduction in direct human labor costs. The autonomous machines involved in the process are commercially available in various forms. Among those machines, industrial robotic arms are the most commonly used leading to many automated solutions. The products produced or processes performed with these machines are highly repetitive with little to no variation. Robotic arms are involved in various applications such as painting in automobile industry, loading/unloading parts, and pick and place operations to name a few.

Industrial robots are available in various configurations and are made up of linkages joined together with motions permitted about each joint and these joints are mechanically stiff and robust. Computer controlled electric motors and hydraulic/air actuators are typically used to control the movements of these joints. The links and actuators are the part of robot's structure and are operated together by the controller that allows them to connect to external systems as well. The actuators' feedback makes the controller capable of precisely controlling the speed, position and torque of a joint. Controller also plans the path to be taken by the robot according to the instructions provided by the user while performing trajectory calculations. As such, controller is performing high level control manipulation with limited interaction with environment due to the proprietary software that does not allow user to modify codes for adding additional intelligence. Sensors may be used to provide feedback from the environment to the controller using different

forms of interfaces that gives the user some tool to interact by providing the operational control and safety. For example, inclusion of vision cameras may allow the controller to avoid collisions with the obstacles in the path of the robot. Likewise, sensors of various kinds can make the robot vigilant to operate around human workers safely.

1.1. Motivation

The motivation for this thesis comes from observing a real-world problem in a manufacturing industry. Various designs of wooden building trusses are produced by a local truss manufacturer from parts being cut using computerized saw to manually assembling the truss. The designs vary and can be multiple quantities of a single truss or custom designed truss for a specific application depending upon the intended use and customer requirement as shown in figure below. A single truss can have four to thirty parts that are of varying lengths from three inches to twenty feet. The trusses are cut by computer numerical control (CNC) saw at the manufacturer's facility. Parts of various trusses are presorted prior to being sent to the saw by the Computer Aided Design (CAD) provided by the manufacturer. The parts are cut in random order to optimize the stock usage. Saw software optimizes the cut order and user cannot alter or modify the cutting sequence.

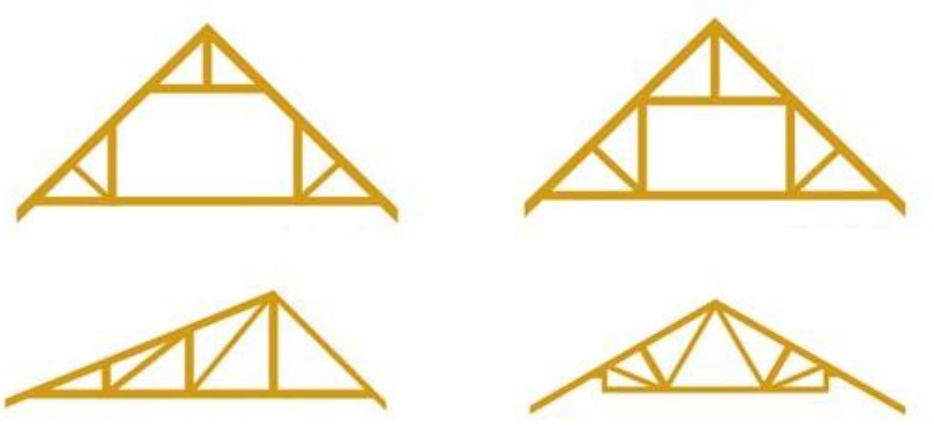


Figure 1: Building Trusses
Source: DIY Doctor

Figure 2: Placement of identical wooden trusses to form a roof where multiple trusses of the same design are joined together to make one complete structure.

Figure 2: Truss Placement Elaboration
Source: Vision Development

The CNC saw executes the part program directly from the CAD software and the part program continuously varies based on the manufacturers needs on a single day. The output from the saw's optimization software is not in an easily readable format. First, stock is loaded into the saw through an automated mechanism. Saw software optimizes the parts' cut order for a batch of trusses and cut parts come out of the saw through a gravity bin. The CNC saw automatically generates a label consisting of letters and numbers that get printed on the trusses. A sample printed truss member is shown in Figure 3.

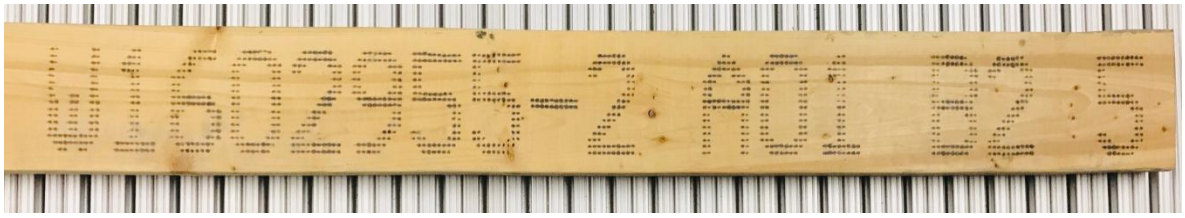


Figure 3: Sample Labeling on a Truss Member

As cut parts drop in a gravity bin in random order, they are to be sorted into various bins and grouped. Parts belonging to a single truss are loaded into a single bin so that the assembler can proceed with the assembly without having to look for parts from a large pile of truss members. Due to non-uniformity and wide variations in sizes of the parts, the traditional method of stacking in a bin is not suitable for this application. Parts will be allowed to overhang the width of the bin.

Currently, one or two workers pick the parts and sort them manually into several bins. They visually read the label and compare to a printed workorder that shows the parts that will make up a truss. A single bin will contain all the parts belonging to a single truss. The process of picking and sorting is laborious and a perfect candidate for automation. Part sorting and stacking is done manually in the current process and parts are placed such that the whole stock remains stable. Once all the parts are loaded in a bin, it is taken to a different station where stacked parts are removed from the bins and assembled together to form a truss. A simulated model of the setup is shown in the figure below.

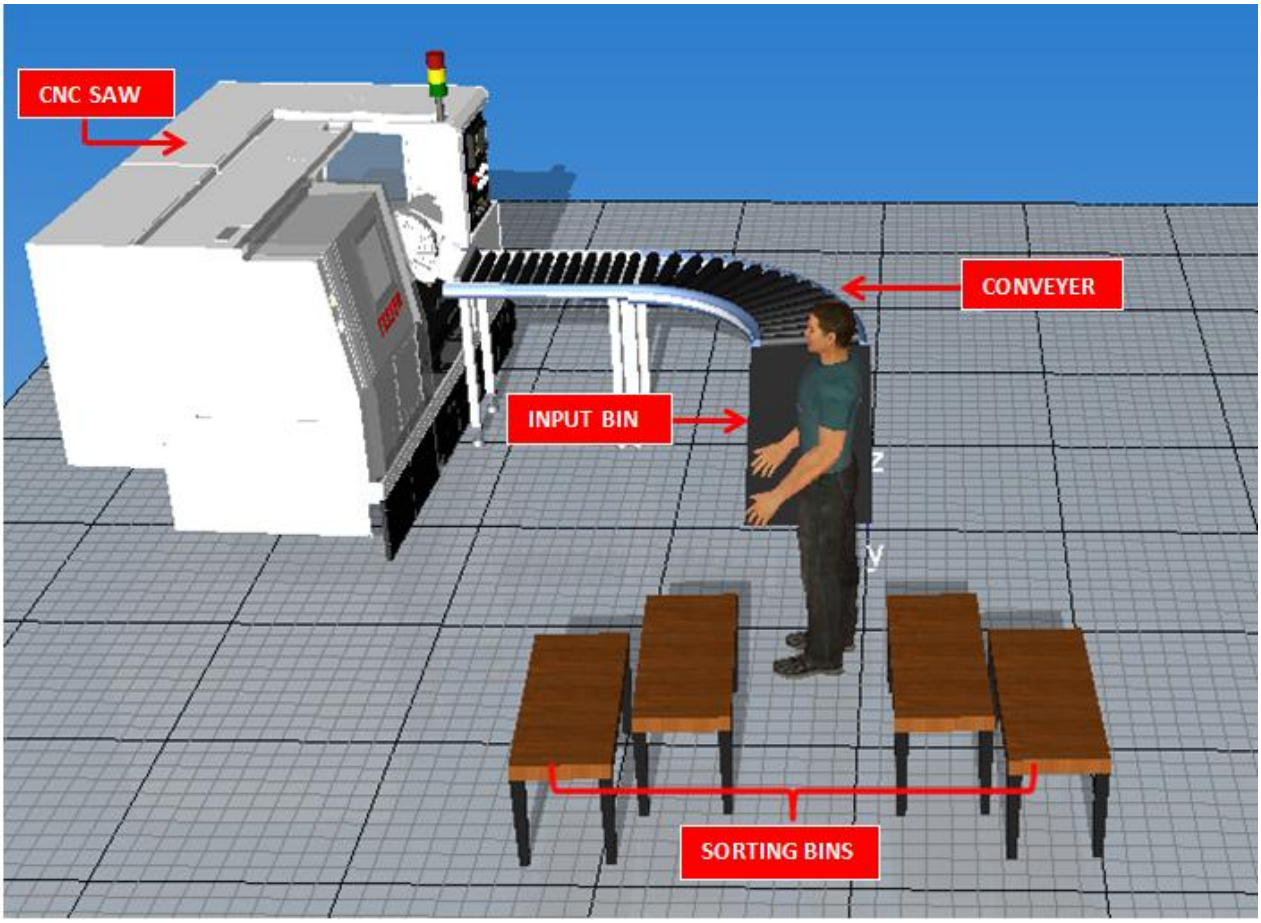


Figure 4: Manufacturer's Factory Set-up Model

The manufacturer has replaced the previous saw with a newer one that cuts parts at a much faster rate to meet increased demands. Manual sorting and packing however is slowing down the process. As such, packing done by the workers is the bottle neck and the manufacturer expressed an interest in developing an automated solution using a robotic arm to speed up the production.

Due to large non-uniformity in sizes of the parts, this problem does not fit under the normal bin packing methods. It is a multi-objective problem where picking, sorting and placement of the parts have to be done in the bin that meets several objectives. The parts need to be placed maintaining stability of the pile as well as consider sequence of removal in the assembly station.

Thus, the focus of this thesis is to develop a strategy that will involve modelling the process with a primary focus on using a robot to pick/sort and stack truss members. Robot interacts with the parts and should pick and place them in the bins meeting all of the above objectives. A number of constraints will be considered. These are further discussed under problem definition in section 3.2.

1.2. Bin Packing & Palletization

In automated production, parts are produced at much higher speeds. Parts produced need to be packaged or sorted if further processing is required. Packing different items in bins is the most vital resource allocation issue in industrial sector especially in manufacturing industries while it also has importance in supply chain and distribution systems. Packing problem occurs in various situations such as cutting, scheduling, loading as well as in planning activities. Applications that deal with storage problem are named as bin packing problem. It involves packing the items in bins or containers with the aim to use minimum number of bins. Different names are given to this problem in literature such as strip or bin packing problem, pallet loading problem or palletization, nesting problem etc.

Three types of classifications are presented for bin packing problem: single bin packing problem (BPP) where different items are assigned to identical bins, multiple bin packing problem (MBPP) in which weakly heterogeneous bins are considered and residual bin packing problem (RBPP) that deals with highly heterogeneous bins [2]. Considering the properties of bins and packing items, each of these problems can be one-dimensional, two dimensional, or three dimensional [3]. Optimal packing saves resources and cost by properly utilizing the bin space. Along with the resource allocation issues, bin packing or palletization problem is also studied to analyze the

impact of different packing methods on the packing efficiency. Various packing algorithms are implemented to increase the packing efficiency of the stacked items and to have an improved palletized system [4]. Development of an exact method that optimally packs the items in bins is not possible. As such, different approximation methods are developed for the bin packing problem. A brief overview of such methods is described in Chapter 2.

An extension to bin packing or palletization problem is the use of industrial robotic arms to automate the process. Manual programming of robots using teach-pendent is time consuming and provides limited solution. An alternative solution is to develop and implement a mathematical based solution using bin packing techniques that interacts and generates automatically paths that would be otherwise generated using manual programming. Stacking the goods into pallets using robots has many advantages in terms of cost and increased productivity. Bin packing algorithms and methods are now being developed by taking the robot's constraints into account as well. Different feedback systems are also used to keep track of the system performance [5] [6].

1.3. Programming of Industrial Robots

Industrial robot's controller is usually not an open source device and a very limited access is provided to users by the manufacturer. Only interface available to the user is a teach-pendent, where robot path can be programmed manually.

The path created using teach-pendent is fixed and any changes to the behavior of the system are to be accommodated by re-writing the code. In situations where variations are present in the system, fixed programs using teach-pendent cannot provide a solution. Small variations can be dealt with multiple pre-programmed paths but it becomes less ideal for large variations. Also, it is impractical to program manually when one is dealing with a large number of variations in paths

required by a widely varying operations. In the local truss manufacturer's production facility, approximately four thousand or more parts are produced in an eight hours shift everyday. The number of shifts increases during summer months due to higher demand for trusses. Also, truss designs may change every day depending upon the requirements. To manually program the path for each part every other day considering all the constraints using teach pendent is humanly impossible. Also, placement position of the parts will not remain the same every time if stability and stacking of the parts are taken into account. It would be ideal to provide an offline solution to such a bin packing problem. Intelligence will be considerably enhanced if user can automatically program path files without having to program using teach pendant that increases the versatility of the machine to adopt to dynamic environment.

This thesis develops a sorting methodology for stable placement of each part using a robot. Proper placement sequence must also be followed to provide the workers an ease of removing the parts for later assembly. The process modelled in this thesis cannot be accomplished by traditional programming of robots. A more effective solution using modeling and simulation tools will be provided. Such work has not been done before. So, the thesis goals are described as follows.

1.4. Problem Statement and Thesis Goals

Past Palletization methods reported in literature do not consider situations when parts are larger than bin boundaries. A situation unique to the problem considered in this thesis where-in stacking the parts with a certain sequence in mind has also not been dealt with. This thesis aims to provide bin packing solution for palletizing the oversized truss parts with full stability by allowing parts to overhang in bins. Also, palletization method must stack the parts to decrease the depalletization and assembly time. Following are the goals of this thesis.

- Develop a palletization algorithm that can pack compactly the over-sized parts by exceeding the bin boundaries. The algorithm must be able to stack parts of multiple trusses in different bins based on their parent truss assembly.

- Develop a methodology for stacking the parts in the bins that minimizes the space within the bin as well as stacks the parts considering sequence that helps ease depalletization of the parts for the assembly process.

- Simulate the bin packing method in a virtual environment to verify the performance of the algorithm.

- Test and implement the methodology in a laboratory environment with scaled parts to confirm the effectiveness of using a robotic arm.

1.5. Organization of thesis

Chapter 2 of the thesis summarizes the literature search pertaining to bin packing and palletization techniques. Chapter 3 introduces the developed methodology to stack the oversized parts that includes sequencing. Implementation and results of the bin packing method including simulations and physical testing is discussed in Chapter 4 and 5 respectively. Finally, Chapter 6 and 7 respectively discuss the results and concludes this thesis with recommendations.

2. Literature Review

Stacking different objects into pallets is a frequently used method for shipping items in bulk. Packing problems has wide range of applications that are of theoretical and practical importance such as loading, scheduling, cutting and routing [7] [8]. Applications that deal with storage problem are referred to as bin packing problem. The problem has importance not only in classical operational setup of loading the items into vehicles but also in various planning activities [9].

Bin packing problem can be dealt with on the basis of storage capacity of bins as well as the size and shape of the items. Generally, the problem is divided into three phases: One dimensional, two dimensional and three dimensional. But the three-dimensional packing problem is the most difficult and computationally complex. There are various variants of bin packing problem and one of them is known as variable sized bin packing problem. In this, a set of items with different dimensions are placed in a finite set of bins and the objective is to pack all the items in minimum number of bins [10].

Bin packing problem is a non-deterministic polynomial-time hard (NP hard) problem. That means the exact solution is too computationally intensive and complex to practically implement. Various placement methodologies have been developed that usually use one of the following techniques: guillotine cuts, simulated annealing, knapsacking and heuristic methods [11] [12] [13].

In guillotine cuts technique, bins are broken down into smaller areas that allow a more comprehensive search within each cut to find an optimal solution and thereby increasing the packing efficiency. Simulated annealing finds globally optimum solutions by using the semi-random search methods. Knapsacking allocates weights and values to the items to pack the most value within the bins that increases the packing efficiency and also allows for object prioritization

[14] [15]. Heuristic methods are used to find the best object placement solutions among all the possible ones. These methods find the solution very fast and it is close to the best one [13] [16]. A higher packing efficiency is produced by choosing best results from various methods, but processing time is also increased to find the required solution.

2.1. Placement Methods

First fit, best fit, next fit or worst fit heuristics can be used to improve placements when placing the items in multiple bins or pallets. In next fit packing method, if an item fits into the same bin as the previously packed item then it is placed there otherwise a new bin is opened and located. [17]. In this method, only the bin where the last item is placed is considered. The previously used bins are ignored. First fit places each item in the first available bin into which the item fits as the algorithm scans the bins from first to last and where ever an item fits it is placed into that bin. [18]. Best fit assigns an item to that bin that leaves the least unfilled space and thus utilizes as maximum space as possible by closely packing the items [19]. On the contrary, worst fit selects that bin which leaves maximum of remaining area after placing an item [16] [20] [21].

The modified versions of the above stated methods such as increasing and decreasing sorting heuristics are also used. In such methods, items to be packed are sorted from smallest to largest or vice versa before placing them in the bins [22]. First and best fit methods produce better packing efficiencies but take more time for calculation. Depending upon the application, increasing or decreasing heuristics can also enhance the packing efficiencies but might not be suitable to use in different applications.

Wall building methods create multiple walls upward up to a certain height on the base of a bin. It is a better packing technique for the items with varying heights. Such method produced

sufficiently better solutions within acceptable times [23]. Various constraints such as geometric and orientation constraints can also be introduced to the packing process before implementing any technique to achieve the desired results [24] [25]. Bottoms of pallets must also be fully supported on the base to simplify the loading process and stability of pallets. Various versions of three-dimensional bin packing problem are considered while applying wall building approach. New constraints are also introduced with their mathematical models built accordingly. An extension to wall building approach is presented in [26] in which the bin is divided into number of vertical layers and then layers are further divided into strips. The strips are then packed optimally as a Knapsack problem with its capacity equivalent to the height or width of the bin.

Layer building heuristics is also commonly used to pack the items in a bin. First, base layer is created by placing the objects on the bottom of the bin. When the base layer is completed then next layer on the top of the previously constructed layer is created. Further layers are created until the height of the container is reached. This method is useful for strongly heterogeneous objects but best implemented for the objects with uniform heights [27]. In [28], two heuristic methods are combined to improve the packing process such that in first method, number of layers are created to place the items into the bin based on the depth of the bin and after that, second method is employed to fill the empty spaces in the bins by heuristically placing the items and thus minimizing the free space. Layer building heuristics is also referred to as container loading algorithm. Various side constraints are observed to make the packing process efficient. The performance of the process is evaluated using different set of instances [29]. Another variant of layer building approach is where layers of identical items are separately created and then packing patterns are generated greedily by loading one layer at a time. Horizontal layer building approach used in [30] uses the same concept of building the layers and then packing them into the bin

according to the established criteria. Guillotine method is also used to orthogonally pack the rectangular boxes in a bin while satisfying the guillotine cut requirement. In such a method, there must exist face parallel straight cuts that can cut the bin recursively into pieces so that each piece contains an object. Cutting the bin into virtual pieces allows more comprehensive search to find an optimal solution and hence increasing the packing efficiency [31].

Block loading technique is also used to fill the items in the bins considering the empty space in the bin. Blocks are created prior to placing them in the bin and are made up of one or more items stacked on top of each other. Then the residual space in the bin is checked that can contain the block. Block loading heuristics based on multi-layer search is also proposed as an efficient way for three-dimensional bin packing problem [26] [32]. In [33], genetic algorithm is proposed to solve the packing problem based on the same phenomenon. First, disjunctive towers comprised of items such as boxes are generated and secondly, those towers are arranged on the bin floor according to the established optimization criteria. Tower creation uses greedy algorithm. Only those tower arrangements that are stable are accepted, meaning that towers containing the items must not fall over and remain balanced. Also, it is considered that all the towers must remain in the bin and parallel to its side walls. Stack building approach or column generation method are also used to describe the block building approach and use the same concept of first creating the blocks of items and then placing them into the bin according to the set criteria [34].

In maximal space placement method, empty spaces are considered as cuboids in the bin and an item is placed in the smallest cuboid that it fits in [35]. It can either be used as standalone method or alongside other techniques to find more optimal placements. Corners formed by the bin boundaries or the existing parts are also used to add new parts into the bin and the method is known as corner method [23]. It also indicates the points where new items can be accommodated

within the residual space of the bin [36] .Corner points are added to directory that the method uses to search. New corner points are added to the directory as new parts are placed and filled corner points are removed. The extreme point concept extends the corner points idea. Extreme points provide a way to check free space within the packing by the shape of objects already in the bin. In [37], extreme point based heuristics is proposed for three dimensional bin packing problem. The heuristic calculates the merit function for the extreme points that are used to place the new box. Similarly, to cater to the feasibility and optimality of the packing, a two level tabu search heuristic is introduced by making decisions at two levels [38]. At the first level, optimality of the packing problem is dealt with while at the second level decisions about the feasible packing are made. Heuristic based on the extreme point first-fit decreasing is used to generate the initial solution.

Different models have been presented to pack the items as the definitive aim is to pack them into minimum number of bins. Various constraints are also introduced to achieve the desired goal. A buffer with constant size can also be used to store the items temporarily before packing them into the required bin. In [39], a bin packing model is presented with a buffer that acts as a temporary storage space .While packing; the objects are either packed into the bins directly or stored temporarily into the buffer before placing them into the desired bin. Storage in buffer can help achieve the efficient packing while ensuring the compactness of the objects. When an object is packed into the bin it cannot be placed back into the buffer and a bin cannot receive the new objects once it is closed.

In [7], different metrics are defined on the basis of various factors. Each metric is given some value and then all metrics are combined together as a heuristic for packing the items into the bin. Some of the metrics include: connections below that measures package overlap. The maximum pressure on top calculation determines the pressure on a box exerted by the boxes on a top of a

certain box. Centre of gravity metric measures box stability. Stack height and box spacing metrics determine the highest point in the pallet and spacing between the boxes respectively. With different weights or thresholds applied, the quantitative metrics are combined to generate an overall measure of goodness. Likewise, online bin packing is also used to solve the bin packing problem where decisions are made immediately for the placement of long sequence of incoming items of variable sizes into the bins. Heuristics is created based on many parameters which are combined to make various potential decisions. Heuristics selects the highest value option while taking the decisions corresponding to a policy [40].

Hybrid packing algorithms are also used to tackle the packing problems. Heuristics combined with variable neighbor search method as implemented in [14] [41] is also used to improve the packing of homogeneous items. Particle Swarm Optimization (PSO) method based on population coverage results in a solution with high quality. During the search process, it explores various spaces of candidate solutions efficiently. It is also very interesting with regard to its simplicity and the number of variables used as compared to other methods. Mixed-case palletization where some of the decision variables are restricted to be the integer values at optimal solution is also considered as a three-dimensional bin packing solution with various side constraints such as weight, height, stability and order line constraints. Side constraints can either be used directly or to narrow the search space. Mixed integer programming formulation can be applied to mixed-case palletization that enhances the density of the bottom layer and maximizes the compactness of the whole pallet to keep the stability of the top layers [42] [43] [44]. Simulated annealing, a combinatorial optimization technique is also used to find optimal solutions for bin packing. Extensive simulation results shown in [45] reveal the high quality optimal solution obtained by this method.

In the modern era, automation is implemented in the industries and hence the use of robots. Bin packing or palletization can be effectively done using the robots with high efficiency. Industrial robots are commonly used for pick and place operation in industry such as painting, placing objects to and from the conveyer belt as well as palletization with speed and accuracy. Robots used in industry should be able to work skillfully in limited space with speed equal to or exceeding to that of humans [46] [47] [48]. Use of robots for packing the items into the bin in real time scenario require one to consider its parameters and limitations along with bin packing techniques. It puts extra constraints while using robots to pack the objects into the bins such as avoiding collision with the previously packed items [49]. Singularity of robots needs to be taken into account as there might be some points or positions that are not reachable by the robot in real time.

Various transformation techniques such as Denavit-Hartenburg (DH) parameters of the robot can be taken into consideration for the pick and place operation that represents the kinematic structure of the robot. Different software tools are also interfaced with the robots to help perform the desired operations. Picking and placing an object using a robot also requires one to choose the geometric location from where it can be picked. Picking an object from its center of gravity is the most common way used in robots that helps in increasing the precision of the operation and also improves the accuracy and safety of the required task [50].

2.2. Verification and Assessment

Various methods have been introduced previously to solve the bin packing problem or palletization. Regardless of the method, verification must be done to make sure that the chosen method functions as desired and without any errors. Verification can include the simulation of the method, graphical model or physical testing as well. If the robot is involved in packing process,

then extra safety measures are required to be taken. For three-dimensional bin packing problem, collision and stability of the parts need proper attention. Intersection of the objects with each other leads to the errors due to collision while stability issues are produced by placing the parts such that there is a risk of falling [51] [52]. Graphical simulation can be used to resolve and verify both the issues and various previous works have efficiently described visual simulation techniques for verification [53].

Simulation tool based on C++ was used by Lim in [54] to verify the packing of parts into the bin. Robotic arm was simulated for the stacking of parts. Using this simulation tool, different techniques were focused to optimize the trajectory of the robotic arm to avoid collisions between the arm and already placed parts. But stability of the parts was not considered while placing them. Reachability of the robot to all required points and positions can also be determined by using the simulation tool and it is also effective to check singularity in the motions of a robot.

Pallet viewer software is used to calculate and display metrics for stacking of items in [7]. The Pallet Viewer program is advantageous for evaluating the procedure of creating a pallet. It assesses the stability of the stack to see whether it can remain stable after construction as the placement of boxes on stack is done using the matrices evaluation technique. Order of the boxes as they are put on the stack is also an important aspect of palletization and was also considered in pallet viewer. Motion planning can also be observed using the software to check whether it is making it difficult or impossible to stack. Additional information such as intersection errors, loading order errors, maximum allowed pressure etc. is also displayed and analysed.

A robotics simulation tool based on video game engine, ‘USARsim’, having three-dimensional physics environment has also been used for the verification of palletization methods. Performance

of the proposed bin packing methods can be assessed using the simulation tool and various problems such as parts shifting, falling or tipping can also be visualized. In [54], authors described the integration of Robot operating system (ROS) and USARsim to simulate the robotic arms and mobile robots more accurately. USARSim and ROS both were applied as a verification method for bin packing technique with robotic arms in [53] [55]. Matlab is also commonly used to implement and verify bin packing techniques. Pick and place operation using robotic arm is frequently utilized to fulfil the requirement. Centre of gravity of a part is of utmost importance while picking up the parts. Matlab is also employed to find and guide the robot to move to the centre of gravity location of a part for pick up [50].

2.3. Stability Assessment

Stability of the stack is an important aspect of bin packing and palletization. Every method includes some criteria to determine the stability of the parts before placement and without it, stack arrangement will collapse and whole purpose of the packing will be lost [14]. So, in all physical stacking or palletization, it is ensured that the resulting arrangement of the stack will remain stable without shifting or falling. Certain vertical and horizontal constraints are also introduced for vertical and horizontal stability of the parts. Vertical stability constraints avoid the parts from falling on the floor of the bin or over each other while horizontal stability constraints prevent the parts from shifting [56] [57].

The main focus in the previous research was mainly on two things regarding stability: support of the base area or the part's edges. Prime emphasis of the area support is for such applications where structural support is provided by the parts or its contents while edge support is mostly used for the applications where structural integrity of box is considered for stability rather than the

contents of the boxes to support weight [27] [33]. Full area support for the placement of the parts is considered by many authors. Corner support method is also used for stability of the parts. In [58], corner support method is used which require all four base corners of a part to be supported for a valid placement. Other methods have used different criteria's for stability. Support of each part on two opposite edges by the parts below it or a certain percentage of the area must be in contact with the lower parts are one the measures of the stability used in [55].

Similarly, stability criteria developed by Carpenter and Dowsland was based on three key factors: jagged or straight cuts should not cut more than a specific maximum bin width and length, a part's base should be in contact with minimum of two parts below it, and the part's base should have a certain percentage threshold of its area that is in contact with the layer below it [59]. The first criterion avoids the packing method of guillotine cutting from making the separate stacks. The second criterion ensures that parts will provide support mutually and interlock with each other while the third criterion makes certain that part is supported on most of its base area. The third criterion proposed by Carpenter and Dowsland that includes certain percentage of area support is mostly used to check the stability of the parts. A summary of findings from various literature reviewed in this section is shown below.

Table 1: Summary of Literature Search

Paper	Year	Packing Method
Geunsoneg Jung, Xu Jing, Jaehyuk Cha, Sungjae Han **[17]**	2015	Next Fit
Taha Ghasemi , Mohammad Reza Razzazi **[18]**	2016	First Fit
Lei Huang , Zhong Liu , Zhi Liu **[19]**	2014	Best Fit
Aristide Grange, Imed Kacem, Sébastien Martin **[20]**	2018	Worst Fit
E. Lopez-Camacho, H. Terashima-Marin, P. Ross and G. Ochoa **[16]**	2014	
G. Stavrinides and H. Karatza **[21]**	2011	
Stephen Balakirsky, Thomas Kramer, Frederick Proctor **[7]**	2011	Metrics Combination
Liu Shenga, Zhao Hongxiaa, Dong Xisonga, Cheng Changjian **[24]**	2016	Wall Building
Hongteng Wua, Stephen C.H. Leungb, Yain-whar Si, Defu Zhanga, Adi Lin **[26]**	2017	
Mhand Hifi, Labib Yousef **[41]**	2018	Hybrid algorithm
Roberto Aringhieri, Davide Duma, Andrea Grosso, Pierre Hosteins **[14]**	2018	
Samir Elhedhli, Fatma Gzara, Yi Feng Yan **[43]**	2017	Mixed Integer Programming
F. Parreno, R. Alvarez-Valdes, J.F. Oliveira, J.M. Tamarit **[44]**	2010	
X. Zhao, J. A. Bennel, T. Bektas and K. Dowsland **[27]**	2016	Layer Building
Ana Mouraa, Andreas Bortfeldt **[29]**	2017	
Rommel Saraiva, Napoleao Nepomuceno, Plácido Rogério Pinheiro **[30]**	2015	
David Pisinger, Rasmus R. Amossen **[31]**	2010	Guillotine Cutting
S. Takahara **[23]**	2008	Corner Points
Teodor Gabriel Crainic, Guido Perboli Roberto Tadei **[37]**	2007	
Teodor Gabriel Crainic , GuidoPerboli , RobertoTadei **[38]**	2009	Tabu Search
Defu Zhang , Yu Peng, Stephen C.H. Leung **[32]**	2012	Block Building
Cedric Joncour , Sophie Michel , Ruslan Sadykov **[34]**	2010	
K. Fleszar **[12]**	2012	knapsacking
Y. P. Cui, Y. Cui and T. Tang **[11]**	2015	
S. Hong, D. Zhang, H. C. Lau, X. Zeng, Y.W. Si **[13]**	2014	

2.4. Overhang

All bin packing methods discussed in the literature described different ways to efficiently pack the items in the bin or pallet but none of these algorithms allowed the parts to exceed the bin boundaries. Parts were not allowed to overhang either and they were strictly constrained to lie within the bin boundaries. When these constraints are taken into consideration, the overhang is defined as the percentage of the area that is not supported by the lower parts including the gaps between the supported areas [23]. But it is quite possible to place the parts exceeding the bin boundaries through overhang and parts larger than the bin itself can also be placed in the bins by allowing the boundary overhang while overall efficiency of the packing may also be increased as shown in the Figure 5.

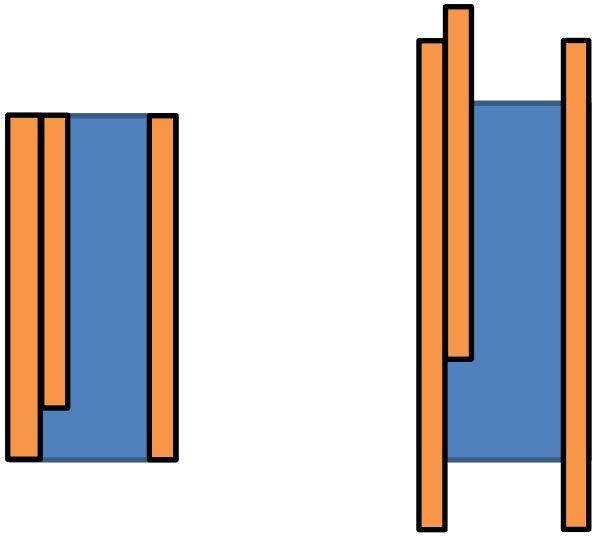


Figure 5: Without Overhang (Left) vs. With Overhang (Right)

Secondly, in the reported literature, parts are packed into the bins by packing them with each other as compact as possible without considering the depalletization process. In some cases, the packed items are required to be removed from the bins to go to another station for further processing. Random arrangement of the goods might not be a good idea in that scenario. Items may need to be packed according to the next required task and hence depalletization and overall

processing time will be reduced. This aspect was considered in this thesis and is an important aspect to be included for automating truss members stacking outlined in the Chapter 1.

3. Methodology

3.1. Palletization Method

The developed palletizing method will have the ability to determine packing position of each part while checking for stability as they are stacked. The algorithm is divided into two sub-algorithms: the placement algorithm and stability algorithm. The placement algorithm searches for the potential placement locations while stability algorithm checks for the stability of that position where the part is to be placed.

3.2. Assumptions and Design choices

The following assumptions and design choices are considered while developing the palletization methodology.

Extreme Point First Fit Heuristic: A greedy, ‘extreme point first fit’ heuristic method is the chosen strategy for the placement of parts. Points are checked for the placement of parts on the basis of first available space. Algorithm searches for the availability of the space across the bin and where the first suitable stable placement is found, a part is placed there and then algorithm looks for the next incoming part.

No Pre-Sorting: The saw that cuts the part is guided by an optimization software and cuts parts by combining all the parts belonging to a number of trusses with a goal to minimize wastage. The user does not have access to the software. Parts selected by this software for cutting may not be in the sequence for optimal stacking. A greedy algorithm would be the best choice for this application in order to minimize the stock wastage and material usage. Pre-sorting prior to cutting may not give the best stacking strategy and will require a lot of robot’s working envelope. Moreover, this work has no access to the saw’s software and presorting is not a possibility.

Single Bin Packing: Parts are separated into their parent trusses before moving to assembly and hence each and every collection of truss parts must be placed into their own distinct bin. Each truss parts goes to a separate station for assembly and thus one cannot mix the parts of different trusses into one bin. Size of the bins is fixed and can accommodate a maximum of 30 parts and these are open bins. Open bins are considered because the length of an individual lumber can be as long as 20 feet. A closed bin may not be an appropriate choice as parts are taken to the assembly station after placement into the bins where they are to be assembled before shipping. Open bin is also a common practice that lumber industry uses. The sorting and stacking are presently done by two human operators. They use their judgement to stack them stably. However, they do not stack them with any consideration of how they will be withdrawn at the assembly. If they are required to consider sequencing/stacking it may require more stackers than what is currently used or may take much longer for the cut truss members to be delivered to the assembly station.

No Part Rotations: Parts used for placement are narrow and long and need to be placed lengthwise in the bin. Available area in the bin will be greatly reduced if parts are rotated for placement.

Scaling: Parts used for placement can have the length varying between 1 to 20 feet and a scaling factor of 4.4 was used for the experimental verification in the laboratory. The Laboratory cannot accommodate longer pieces and the experimental cell robot also has very limited reachability.

Unity Development Environment: The development and simulation environment used in the research is a video game engine, ‘Unity’, developed by Unity Technologies. It has ease of access and large development community. In particular, the built-in physics engine allows one to analyze

gravitational and frictional effects during simulation and hence a better tool to check the part placement and its stability.

Octopuz Verification Environment: To verify path planning of the robot, 'Octopuz', developed by Octopuz Inc., is used. It is a simulation environment typically used for path planning of industrial robots. Octopuz verifies the path planning of the robot and also generates the path files for the robotic arm. It offers excellent path simulation capability, enhanced developer support and simplicity of integration with the robot.

3.3. Placement Algorithm

'Extreme point first fit' heuristic bin packing algorithm was developed to place the parts compactly into each bin, limiting the amount of wasted space and ensuring that each placement remains stable. Extreme points are the corner points generated by previously placed parts and stored into the directory to check the potential placement for the new incoming parts. First fit heuristic searches for the first available space across the bin to place the new part.

Named parts of a building truss are used for identifying/stacking of truss members in the bins. All the building trusses have a common terminology and various parts of a truss are identified and categorized as shown in Figure 6.

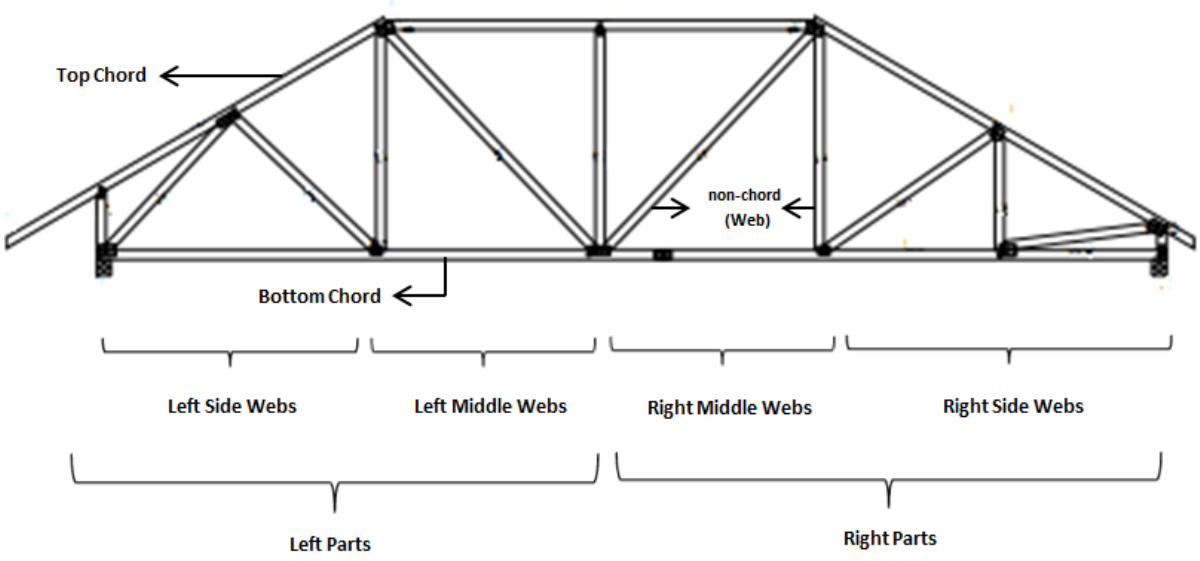


Figure 6: Building Truss Terminologies

To facilitate the assembly of a truss from individual parts, the truss parts are sorted into a respective bin as left or right-side parts and placed into the bin such that left side parts are placed to the left side of the bin while right side parts to the right side of the bin as shown in figure below. Two assemblers are involved in assembling a truss. Left side parts are assembled by one of the assemblers and the right-side parts are assembled by the other.

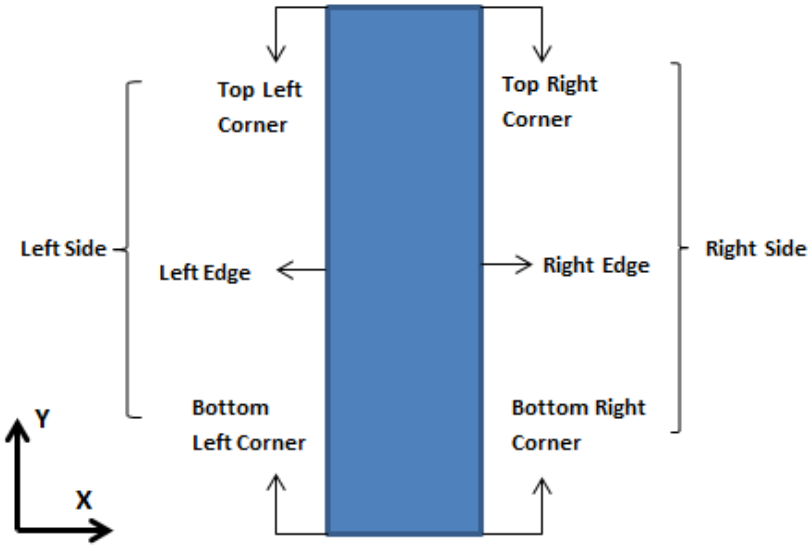


Figure 7: Bin Definition for Part Placement

The placement of parts in the bins must take into consideration of how the individual parts are withdrawn from the bins. Hence the algorithm must consider appropriate sequencing also. Sorting is further sub-divided into two categories: chords and non-chords parts. Chords are the parts of a truss that make the outermost boundary including the base and outer sections and are placed first on the assembly station while assembling the individual parts into a truss. When a part appears in the input bin, algorithm checks to see whether it is a left side or right side part. Upon identification, it then looks at whether it is a chord part or not. If it is a chord part, it is placed either on the leftmost or rightmost edge of the respective bin depending upon the side of the truss such that the X/Y center of that part is placed to the Y center of the bin to maximize the stability as shown in Figure 8. When next incoming part is also a chord part then it is stacked on the previously placed part while algorithm stores the position of the already placed parts to avoid collision.

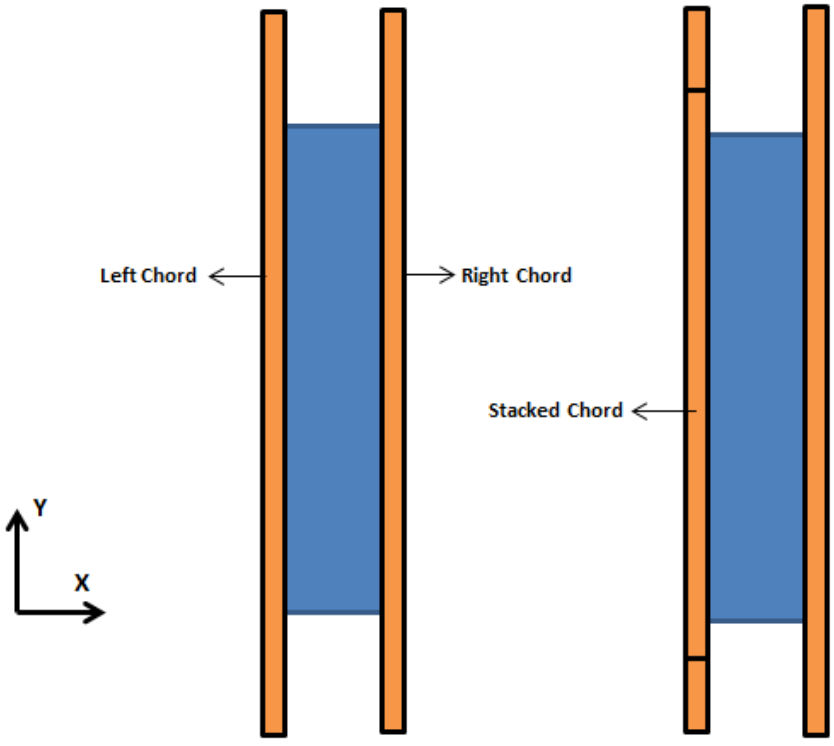


Figure 8: Chords Placement into the Bin

When a non-chord part appears in the input bin, it is placed next to the chord parts near the top left corner for the left side part or top right corner for the right side part if the spot is stable. The algorithm must also check to see for collision. If the placement is valid then part is placed there while maintaining a distance of 15mm along the x-axis of the bin from the adjacent parts to avoid collision. The 15mm gap is used to avoid collision of the parts while robot actually places the parts into the bins in a physical environment and it can be changed if required. The very first part is placed onto the base of the bin such that its 2/3rd length stay inside the bin to have maximum stability of that part. The 2/3rd length criterion is chosen to ensure that the space outside of the bin boundaries is utilized to the maximum, minimizing the amount of in-bin space used and this length criterion can be changed if desired while maintaining the stability of a part. When the next part comes in, it is first searched whether it can be stacked onto the previous part while maintaining the stability. If a part finds appropriate support area, then it is stacked onto that part such that 2/3rd of its length must stay inside the bin as well to have full stability of that part. It is ensured that maximum of three parts are stacked onto one another to have full stability of parts even when a bin is moved to another location after packing. Although, more than three parts can be stacked with full stability onto one another when the bin is stationary it can get unstable when the bin is being moved and the whole purpose of sorting will be lost. This criterion was applied upon recommendation from the experienced sorters in the industry.

If a new incoming part cannot find a stable stackable position onto the already placed parts then algorithm finds the space available next to the already placed parts along the y-axis. If the next available space is such that 2/3rd of the part's length can stay inside the bin while collision and stability checks are validated, it is placed there otherwise, algorithm places such a part near the top left corner for the left side part or the top right corner for the right side part while maintaining

the distance of 15mm from the already placed parts along x-axis of the bin as shown in Figure 9. If a part cannot be placed into a bin then it is placed into a buffer bin as an intermediate storage. Every time when a part is placed into respective bin, the algorithm proceeds to check for a part in the buffer bin. If a part is available in the buffer, algorithm looks for the potential placement for that part into the sorting bins. If a spot is found for such a part then it is placed into a respective bin otherwise it is kept into the buffer while algorithm checks for a new part to be placed next in the input bin.

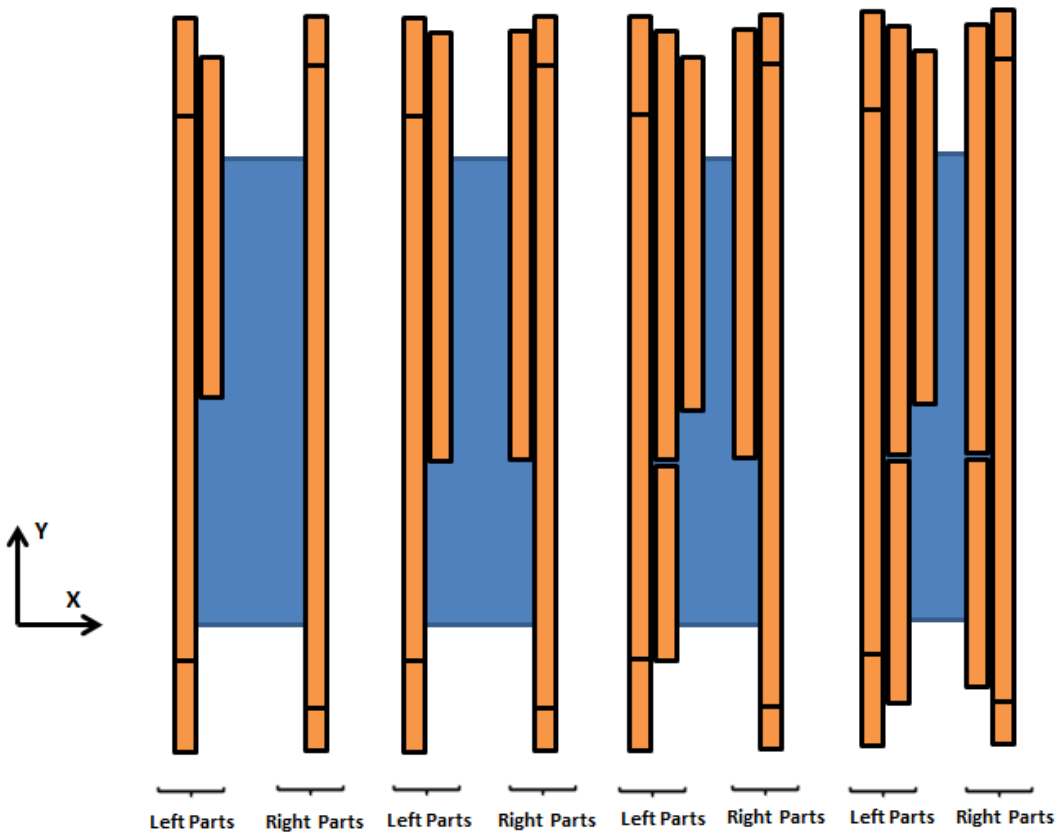


Figure 9: Placement of Chord and Non-Chord Parts

The process flow chart for the overall algorithm can be seen in Figure 10. When a new part comes in, it is first sorted as a left side part or a right side part. Once a choice is made, then algorithm differentiates among the chord and non-chord parts. Chord parts are placed to the edges of the bin

while non-chord parts are positioned adjacent to chord parts as per the developed methodology. Algorithm also checks if any part is in the buffer bin to find its placement into the sorting bins. Before placing a part, algorithm determines the stability of potential location as well as collision. If the position is validated, then parts are placed there otherwise algorithm checks for the next potential location for placement. If no location is found for a part, then it is placed in the buffer and an error message is sent.

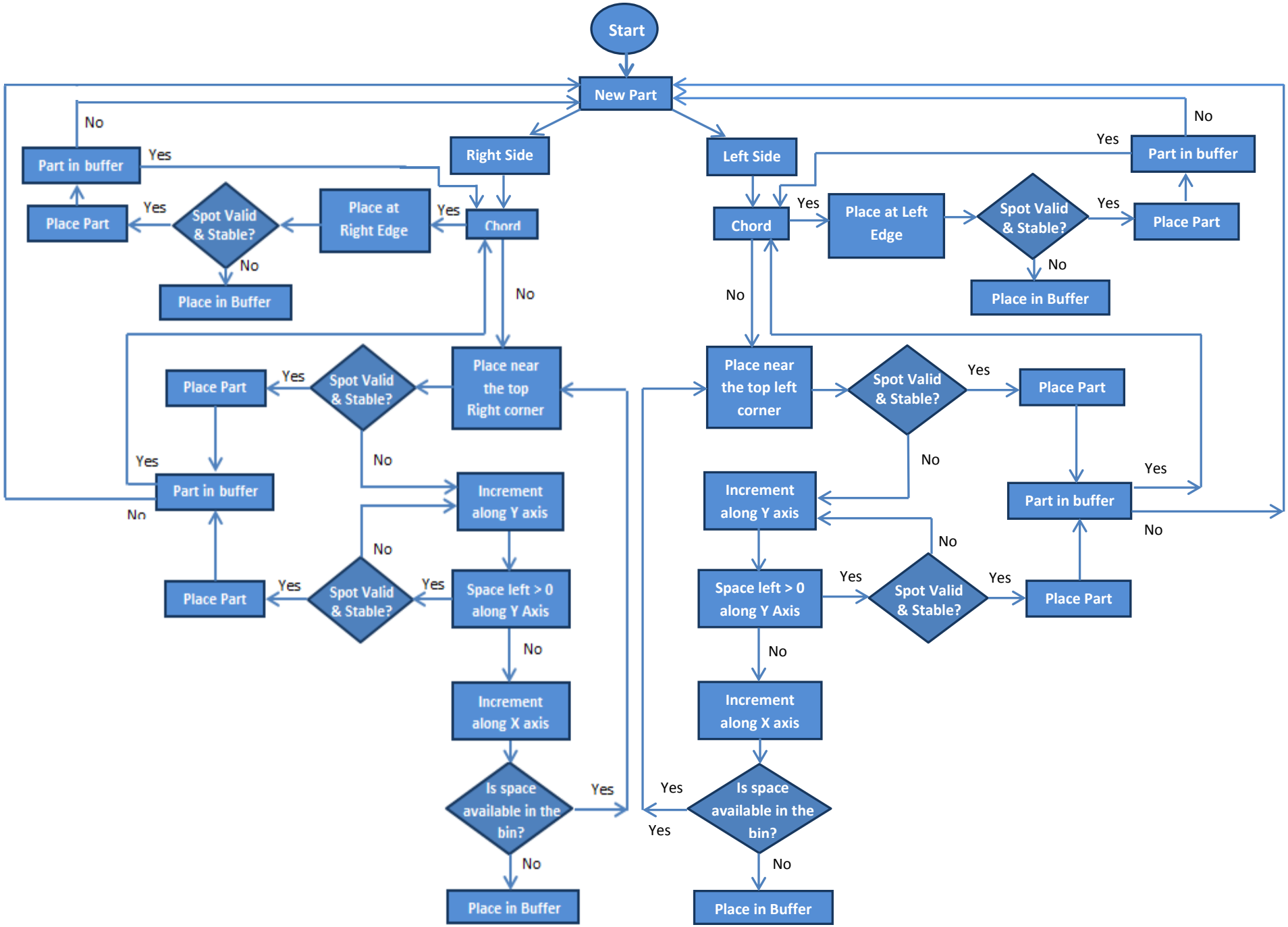


Figure 10: Process Flow Chart

3.4. Color Marking on the Parts

As described earlier, placement methodology is developed to help assembly of truss using the stacked parts in the bins. To verify whether the parts have been stacked properly for assembly, all the parts were manually color coded with a marking to facilitate the verification process. The color coding of truss members was introduced primarily to verify the performance of the developed methodologies. It is not a requirement in the final implementation. The industry may continue to print optically recognizable codes until automated sorters can replace manual operation. Parts used in this research are reproduced from the data available in saw cut files and scaled to 1:4.4 for implementation in the laboratory.

3.4.1. Left and Right Chord Parts

As outlined before, the left and right chords are placed at the left and right edges of the bin respectively. Left is specified with a green color mark while right is marked with a blue color on the parts. All the color markings were manually generated. Chord parts are indicated with a red color. Since, chord parts in a truss can be a top or bottom, a black color marking is used for top chord parts and light green color with yellow boundary box indicates it is a bottom chord. For example, color combination of green, red and black with marking of *LCT1* indicates that it is a first left top chord while blue, red and black colors combination with *RCT1* marking indicates that it is a right chord and is also a first top part to the right side. Likewise, color combination of blue, red and light green with yellow boundary box and marking of *RCB1* represents the right chord as the first bottom part. Marking in alphabets with numbering onto the parts is used to further elaborate the process.

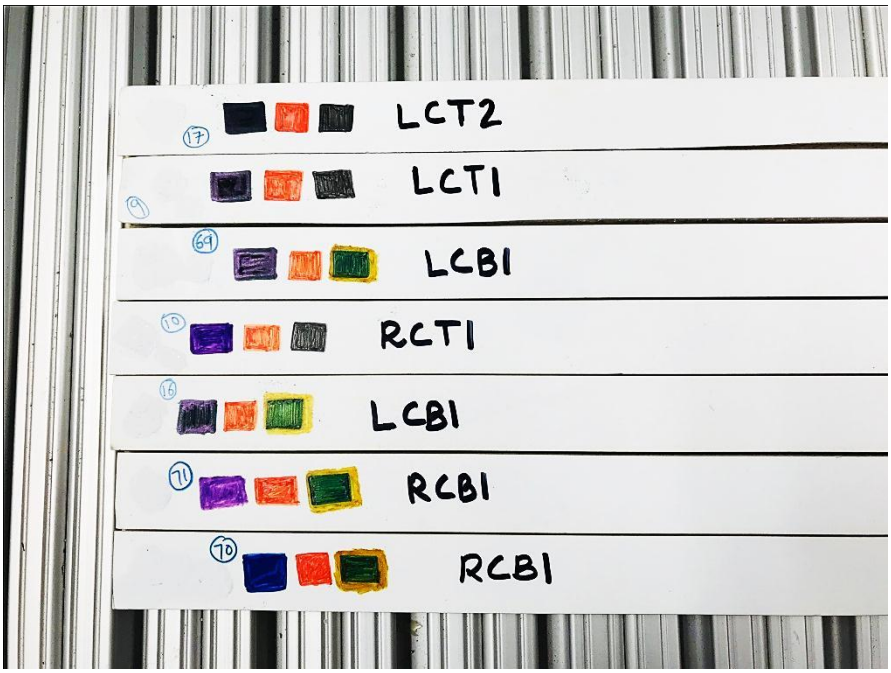


Figure 11: Left and Right Chords Color Marking

3.4.2. Left and Right Non-Chord Parts

Non-chord parts are also color coded and marked with corresponding alphabets and numbering. Non-chord parts are also called ‘webs’ and such parts are further sub-marked as middle and side parts. Orange color is reserved to represent all web parts while left and right webs are differentiated with green and blue color respectively. Middle or side web differentiation is indicated through alphabetic marking with the corresponding number of parts. Middle part is differentiated with alphabet *M* and side part is represented with *S*. For example, green and orange color combination with *LWM1* marking indicates the left web that it is first middle part of a truss. Similarly, blue and orange combination with *RWS1* marking indicates the right web that it is first side part.

Figure 12: Left and Right Non-Chord Parts Color Marking

Figure below shows the color markings on various chord and non-chord parts.

Figure 13: Chords and Non-Chord Parts Color Marking

3.5. Stability Algorithm

A part may shift or fall if does not find appropriate support area below it. Placement algorithm described in above sections places the parts compactly into the respective bins but it does not consider the likelihood that a certain part will shift or fall. Absence of insufficient level of support below a part will affect the stability of new and previously stacked parts and must be avoided.

3.5.1. Stability Assessment

Placement algorithm places the parts as compact as possible to the first available bin location. It is acceptable to assume that a new part can have more than one part below for support. It is observed that the area of the parts below as well as their location namely the distance from the new part's center of mass is very important to determine the stability of the new part. To ensure that the new part is well supported by the parts below, both area and the location are considered to check the stability. Stability factor 'S' that is equal to the contact area 'A' multiplied by its distance from the part's center of gravity 'X' was calculated to analyze the impact of each lower part to the stability of the new part.

$$S = A\,X \tag{1}$$

To further describe the stability factor, let's consider the stability as a single dimension for part P_4, shown in Figure 14, which is supported by parts 1, 2 and 3 below it. The stability factors imparted by these parts would be $S_1 = A_1\,X_1$, $S_2 = A_2\,X_2$, $S_3 = A_3\,X_3$ respectively.

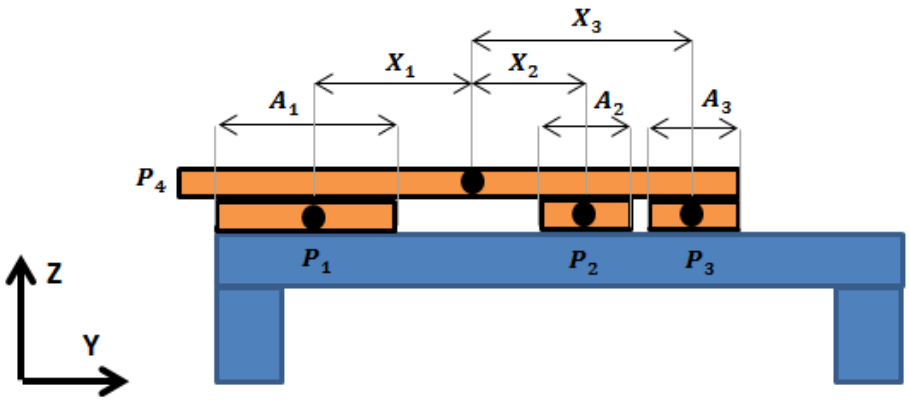


Figure 14: Part Having More than One Contact Below

However, a support area of a part below may cross the new part's centroid. To overcome this, the new part is divided into two quadrants: one to the left and other to the right of the part's centroid. As shown in the figure below, part P_1 is supporting the new part on both sides of its centroid and hence it is in both quadrants Q_1 and Q_2 of the new part P_4. So, P_1 is divided into two segments, one for each quadrant as shown in the figure.

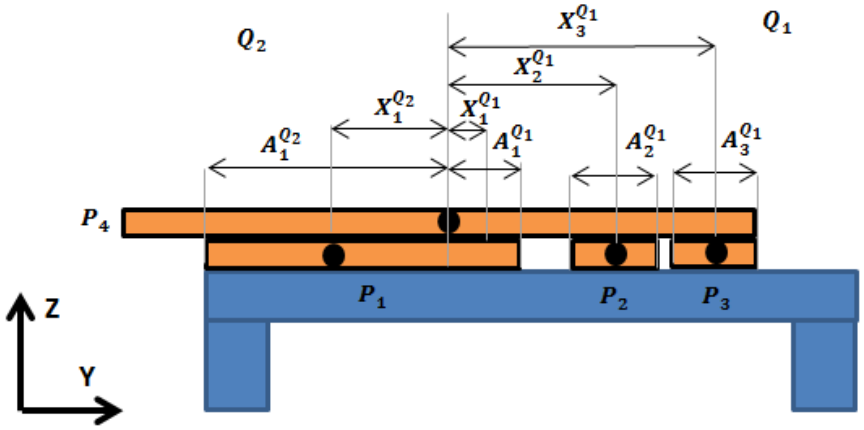


Figure 15: Parts Crossing the Centroid

The overall stability of each quadrant is sum of all the part stability factors it contains:

$$S^{Q_n} = \sum A_i X_i \tag{2}$$

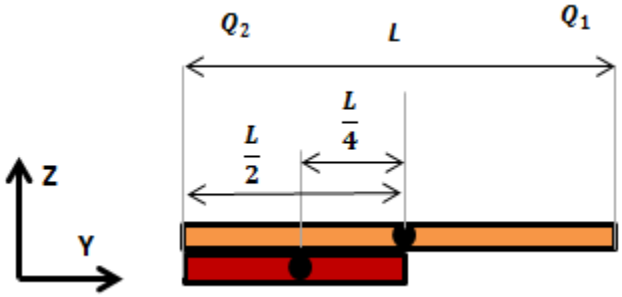
As such, stability of part P_4 for each quadrant can be written as:

$$S_4^{Q_1} = A_1^{Q_1} X_1^{Q_1} + A_2^{Q_2} X_2^{Q_2} + A_3^{Q_3} X_3^{Q_3} \tag{3}$$

$$S_4^{Q_2} = A_1^{Q_2} X_1^{Q_2} \tag{4}$$

where $S_4^{Q_1}$represents the stability of part P_4 in first quadrant while $A_1^{Q_1} X_1^{Q_1}$ is the stability factor of part P_1 , $A_2^{Q_2} X_2^{Q_2}$ and $A_3^{Q_3} X_3^{Q_3}$are the stability factors of part P_2 and P_3 in the first quadrant respectively. $S_4^{Q_2}$ represents combined stability factor of part P_4 while $A_1^{Q_2} X_1^{Q_2}$ is the stability factor of part P_1in second quadrant. It must be noted that, these metrics may be sufficient to find the stability of parts with similar dimensions as part P_4 but such factors cannot be directly compared to varying part sizes. The stability factor shown above is to be non-dimensionalized by dividing it by the maximum possible stability factor for a particular part so that it can be used for parts of varying sizes.

The maximum stability factor for a part is equal to the total length in a quadrant multiplied by half of the distance from the center of mass of the part to the edge as shown in the figure below.



$$S_{max} = \frac{L}{2} x \frac{L}{4} = \frac{L^2}{8}$$

Figure 16: Maximum Stability Factor

The percentage stability factor can be described as:

$$S_{\%}^{Qn} = \frac{S^{Qn}}{\left[\frac{L^2}{8}\right]} \tag{5}$$

This stability factor can now be compared to the minimum threshold to check if the parts below the new part are sufficient to keep it stable. The minimum value of the threshold is determined experimentally and can be changed.

3.5.2. Collision Detection

To pack the parts such that they remain fully stable, stability and collision checks are made before placing a part into an available location. Collision detection is also done along with the stability algorithm to determine the overlap between a part and the parts directly below it. It is an important aspect of part placement to ensure that the new incoming part must not be placed at the location already filled by a previous part. Without the ability to check if a location is already occupied referred to as collision detection, the algorithm would try to place the parts into an area that was taken by the previous parts already leading to collision and ultimately making the whole stack unstable. As well as, collision detection is also used to check that a part must not collide with the parts adjacent to it, ensuring the stability of the parts placed side by side. A process flow chart with stability and collision detection is shown in Figure 18.

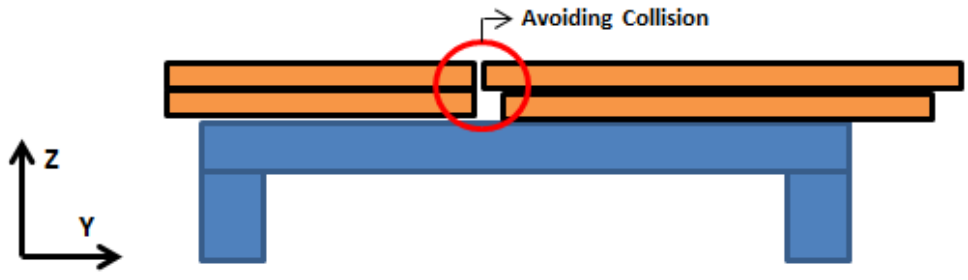


Figure 17: Collision Detection

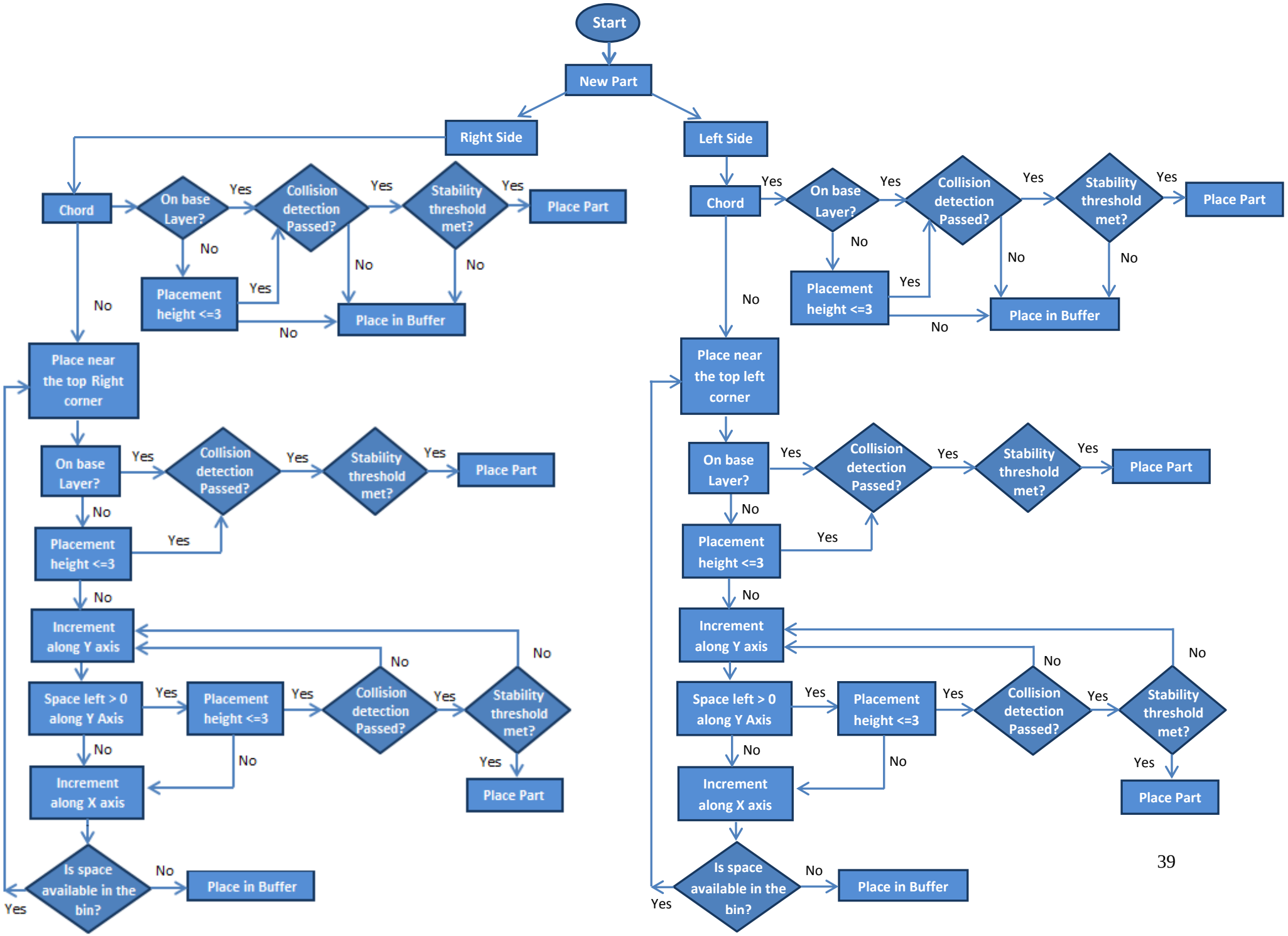


Figure 18: Stacking Algorithm Flow Chart

Left and right side parts are sorted accordingly as shown in the flow chart. The stability assessment is performed with all the parts below the new part as described in section 3.5 and support provided by the lower parts is quantified using the stability factor. The new part is placed only if the overall stability factor is above the threshold. Also, when stability of a part is determined for the potential placement, it is first checked if the part is in the base layer or not and placement is validated according to the stability threshold. X/Y collision detection is also done with already placed parts to avoid any accident. The process is repeated for every part in the bin and placement is validated if stability threshold is met and no collision occurs. Otherwise algorithm signals that the placement is not validated and checks for the next potential location for placement. Based on the developed methodology, a Unity simulation as well as physical implementation is performed on the robot cell. The next chapter describes the complete implementation procedure to access the performance of the method.

4. Implementation

Once placement strategy was developed, it was simulated using the Unity engine and Octopuz software. The resulting palletization algorithm was tested with saw files provided by the local truss manufacturer. The results produced by these tests were analyzed using the assessment methods presented in this section. Physical implementation with scaled parts was also done in the laboratory to validate the performance. The figure below describes the sequence of implementation.

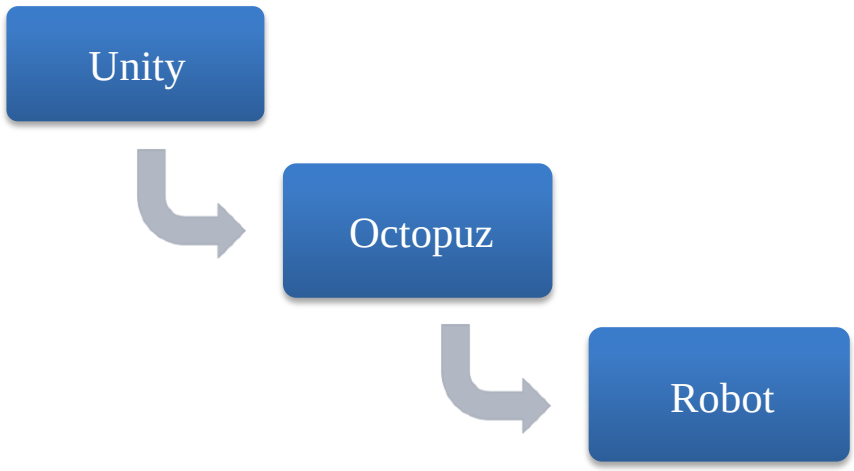


Figure 19: Implementation Process

4.1. Simulation

Unity was chosen to develop the palletization method using C# as a programming language. In-built physics engine that helps analyze the gravitational and frictional effects as well as the collision detection capabilities make Unity a very effective tool for such an application. Packing stability and collision detection of the parts was effectively evaluated using such capabilities of Unity engine. Figure below shows simulation environment created in Unity by considering the

local factory setup that includes an inclined input bin and four sorting bins, one for each truss. A beacon was also added alongside each bin that illuminates once all the truss parts are packed into a respective bin. Buffer bin in the middle was used for temporary storage of parts. This system mimics the factory environment described earlier in Chapter 1.

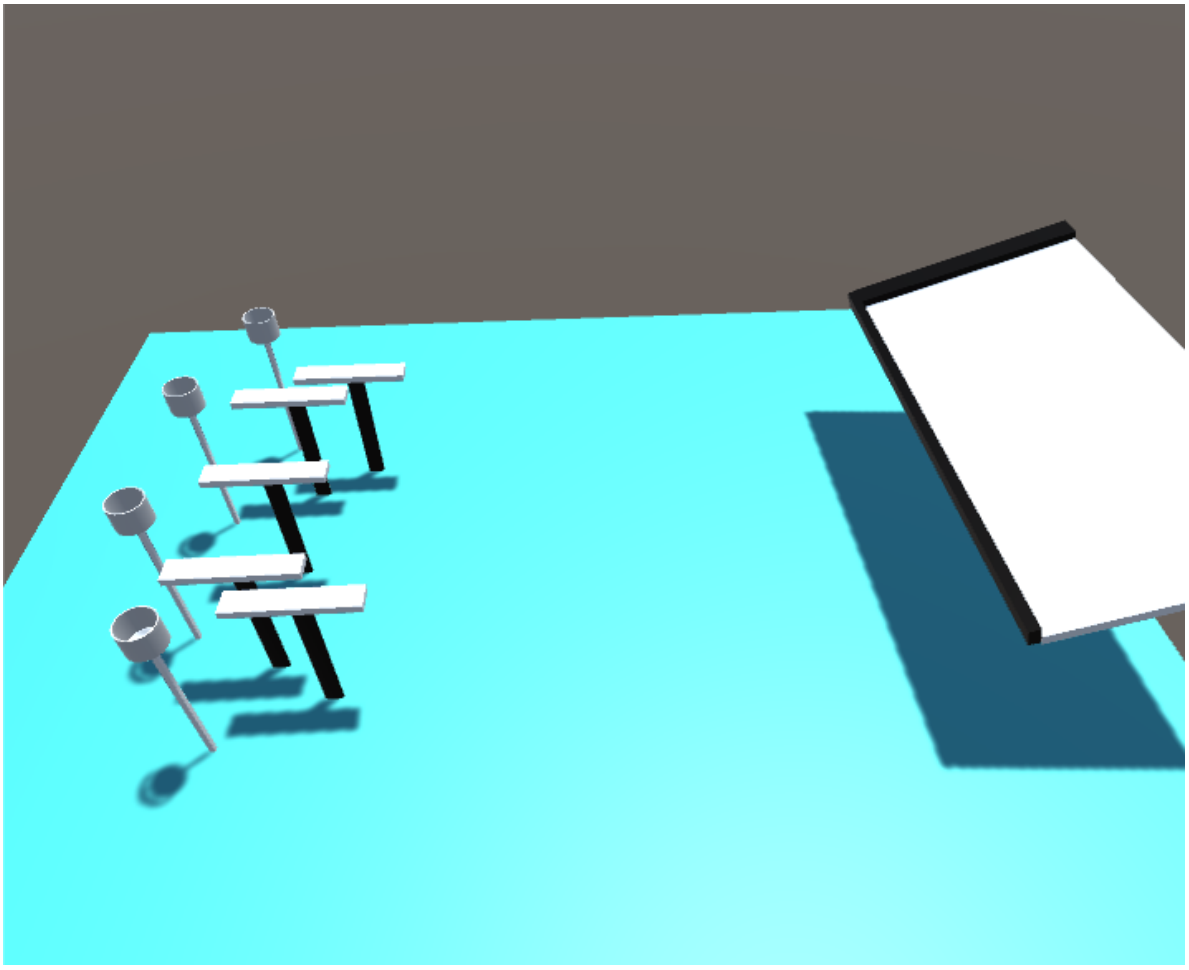


Figure 20: Simulation Environment in Unity

One of the completed stacked simulations of four different trusses can be seen in Figure 21. Different colors were used to differentiate various parts of trusses. Parts appeared in random

order belonging to any of the four trusses in input bin, sorted and packed into the bins according to the methodology. In Unity simulation, visual display was also added to the screen to indicate the progress of packing process. Packing completion status of the truss parts in a bin are also shown on screen along with the illuminated beacon.

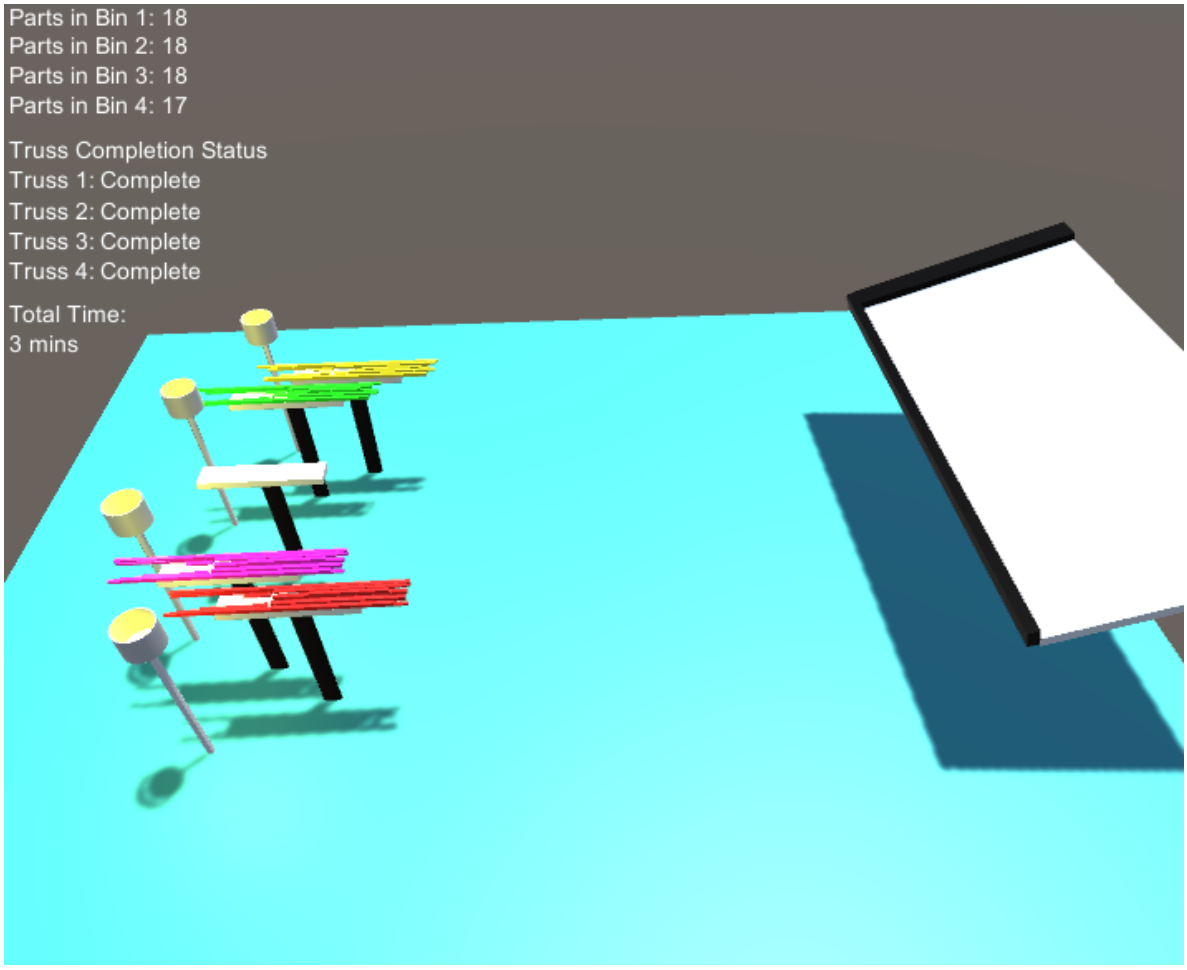


Figure 21: Representative Assembly 1

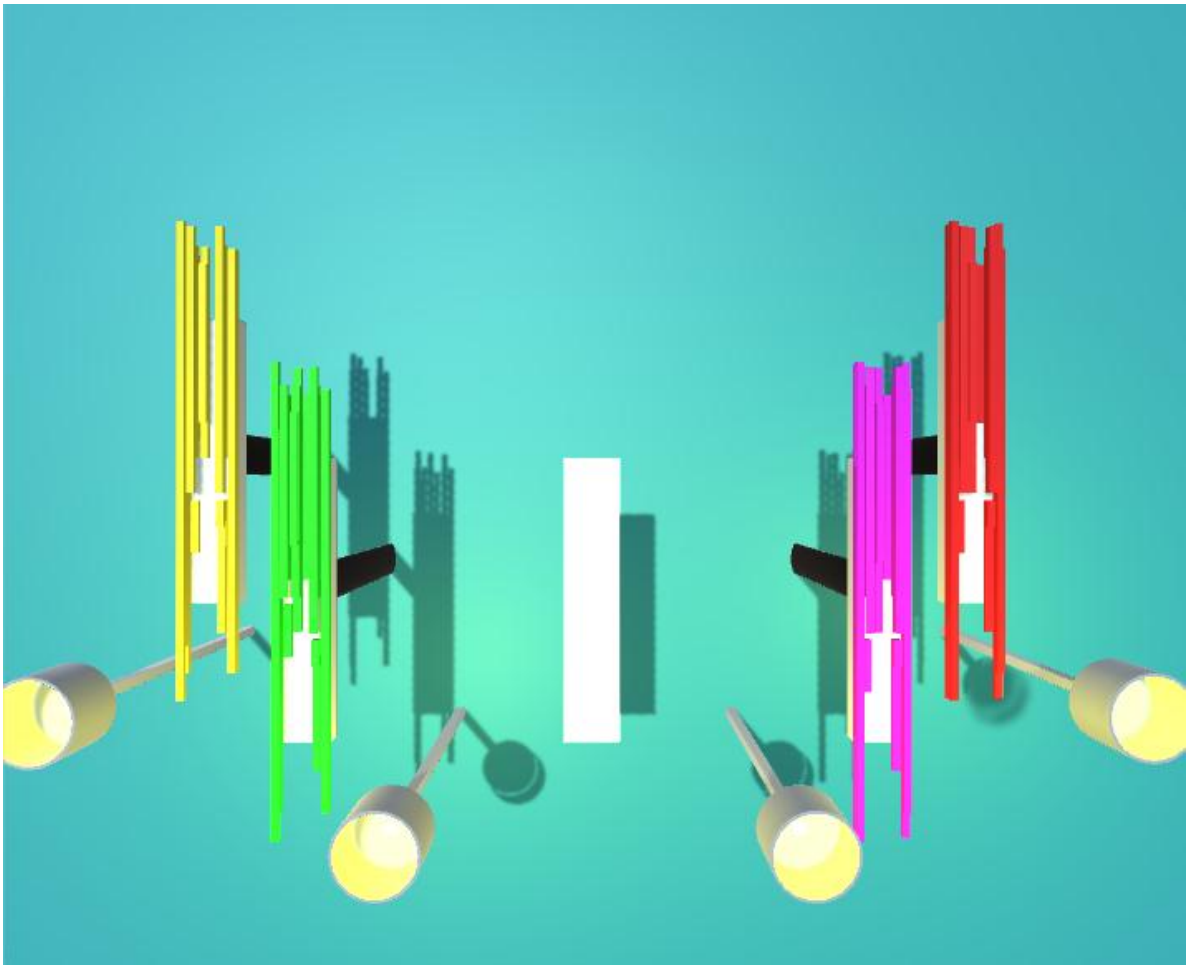


Figure 22: Representative Assembly 2

Before physical testing in the laboratory, path planning of the robot was verified in Octopuz to make sure that the path taken by the robot is valid and did not have any singularities or collisions. Singularity refers to a point in the workspace of the robot where the end effector cannot position itself in the desired location due to the physical limitations of the various joints. The singularity positions cannot be ascertained ahead of programming various path points if a teach pendant is utilized for manual programming. Octopuz was also used to develop the cell according to robot's working envelope. Path taken by the robot for one of the saw files is shown in figure below, where orange lines are the path trajectories.

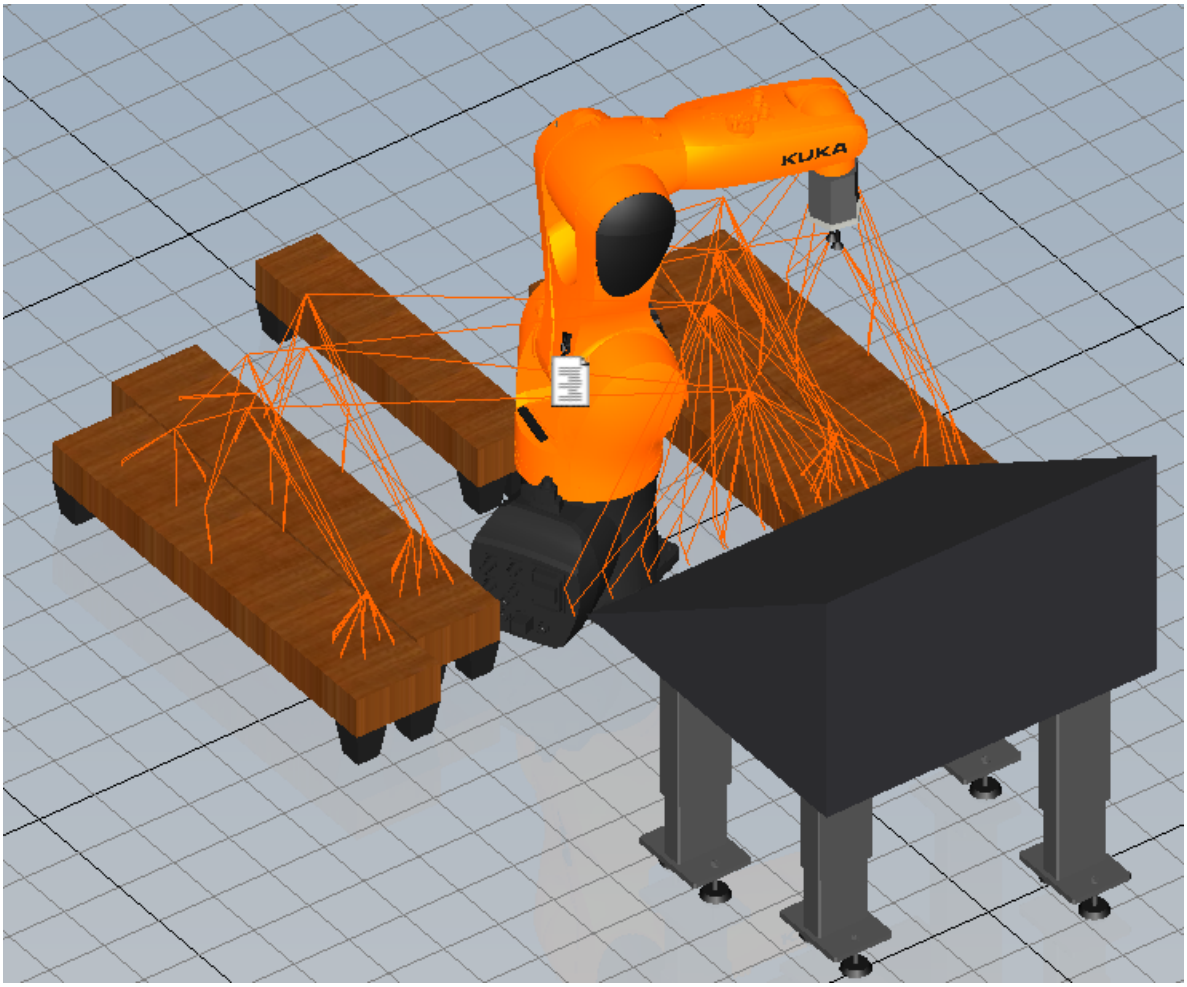


Figure 23: Octopuz Simulation with Path Trajectories

4.2. Assessment Methods

Bin packing efficiency can be measured using various metrics but traditionally volume utilization and packing density are most commonly used. However, these metrics can be misleading as methodology heavily utilizes the space outside the bin while such metrics only measure internal bin usage. To address this issue, additional metrics are also introduced that include the overhang of the parts to measure the overall volume utilization and packing density.

As algorithm sorts the parts into left and right sides as well, all such metrics are further sub-categorized for left and right side volume utilization and packing density separately.

The First, In-Bin Volume Utilization, is the measure of volume usage within the bin boundaries. It is defined as the ratio between the volume of the parts within the boundaries of the bin and the volume of the bin on either side.

$$V_{LTBin} = \frac{\sum_i^n P_{i\,LTBin}}{(\frac{X_{Bin}}{2})Y_{Bin}Z_{max}} \tag{6}$$

$$V_{RTBin} = \frac{\sum_i^n P_{i\ RTBin}}{(\frac{X_{Bin}}{2})Y_{Bin}Z_{max}} \tag{7}$$

where V_{LTBin} and V_{RTBin} are the in-bin volume utilizations of left and right side parts respectively. $P_{i\,LTBin}$ is the volume of left side parts while $P_{i\ RTBin}$ is the volume of right side parts within the bin boundaries. X_{Bin} and Y_{Bin}represents the X and Y extents of the bin. Here, the whole bin area is divided into two equal halves along the x-axis of the bin as shown in figure below; one half is for the left side parts and the other half is for the right side parts. Additionally, open bins were used for part placement that did not have predefined heights and parts can be stacked to the desired height into such bins so, Z_{max} maximum part height in the respective side is used along with the X and Y extents of the bin for calculating the bin volume.

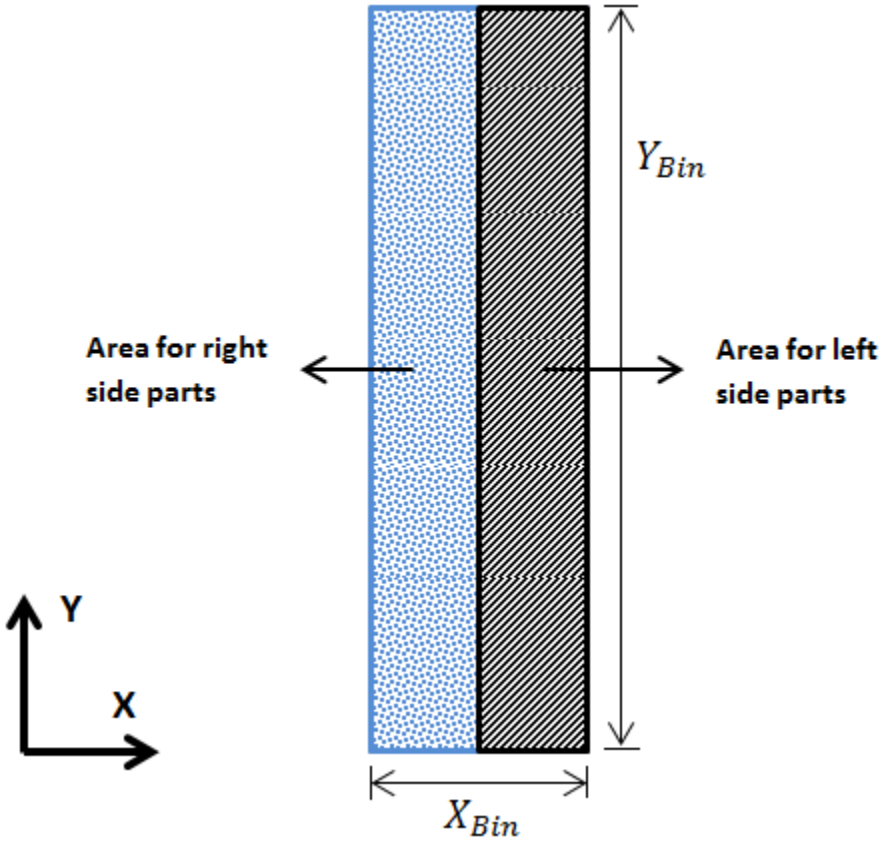


Figure 24: Division of Bin Area

Overall volume utilization is used to measure the volume usage including the overhang of the parts and it is measured as the ratio between the overall volumes of the parts and the volume of the bin for either side as shown in equation below.

$$V_{LT} = \frac{\sum_i^n P_{i\,LT}}{(\frac{X_{Bin}}{2})Y_{Bin}Z_{max}} \tag{8}$$

$$V_{RT} = \frac{\sum_i^n P_{i\ RT}}{(\frac{X_{Bin}}{2})Y_{Bin}Z_{max}} \tag{9}$$

where V_{LT} and V_{RT} are the overall volume utilizations of left and right side parts respectively. $P_{i\,LT}$ is the volume of left side parts while $P_{i\ RT}$ is the volume of right side parts including the overhang. X_{Bin} and $Y_{Bin}$are the X and Y extents of the bin while Z_{max} is the maximum part height, used as a third dimension in calculating the bin volume.

Packing density metrics were also used to measure the volume utilization of parts with and without overhangs. In-bin packing density measures the parts packing density within the bin boundaries as shown below.

$$\rho_{LTBin} = \frac{\sum_i^n P_{i\,LTBin}}{(X_{BinMax} - X_{BinMin})(Y_{BinMax} - Y_{BinMin})Z_{max}} \tag{10}$$

$$\rho_{RTBin} = \frac{\sum_i^n P_{i\,RTBin}}{(X_{BinMax} - X_{BinMin})(Y_{BinMax} - Y_{BinMin})Z_{max}} \tag{11}$$

where ρ_{LTBin} and ρ_{RTBin} are the left and right side in-bin densities of the parts respectively. X_{BinMax} and Y_{BinMax} are the maximum utilized extents while X_{BinMin} and Y_{BinMin} are the minimum utilized extents within the bin boundaries.

Overall packing density of the parts was used to measure the overall parts volume including the overhang as shown below.

$$\rho_{LT} = \frac{\sum_i^n P_{i\,LT}}{(X_{Max} - X_{Min})(Y_{Max} - Y_{Min})Z_{max}} \tag{12}$$

$$\rho_{RT} = \frac{\sum_i^n P_{i\,RT}}{(X_{Max} - X_{Min})(Y_{Max} - Y_{Min})Z_{max}} \tag{13}$$

where ρ_{LT} and ρ_{RT} are the left and right side part's densities respectively with overhang. Here, the smallest bounding box that fits all the parts in either side was considered while Z_{max} is the maximum part height.

4.3. Physical Testing

After verifying the methodology in Unity engine, it was implemented physically in a robot cell in laboratory using an industrial robotic arm. This robotic arm had a reach of 706.7mm with six degrees of freedom. Robotic arm used in the laboratory had one quarter of reach as compared to the full scale model (706.7mm vs 3095mm) to be used in the factory setup. A scaling factor 4.4 was created to reproduce the scaled parts for smaller robot in laboratory by dividing the reachable limits of full scale and smaller robot. The robot cell was first modeled in Octopuz according to the robot's working envelope. The figure below shows the developed robot cell in laboratory that has KUKA KR6 R700 SIXX robot, one inclined input bin, four sorting bins, one for each truss and one buffer bin for temporary storage. The inclination of the input bin was 30° while all the sorting bins had a width of 5.7″ and a length of 30″. These are open bins with no pre-defined heights.

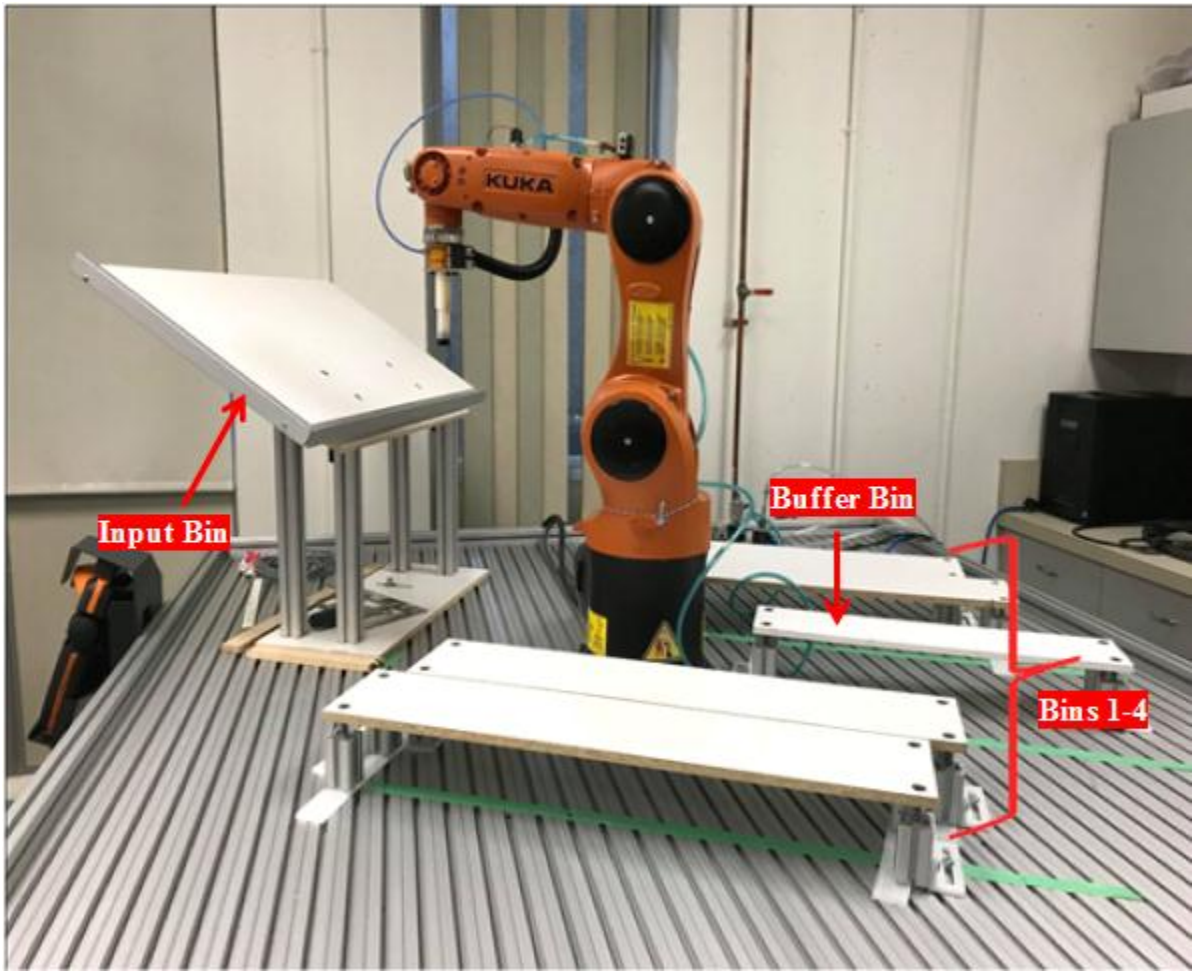


Figure 25: Robot Cell

Path file created by Unity was first simulated in Octopuz to verify the path trajectories that robot took for parts placements in the bins. Validity of all the positions was checked along with singularities or any other errors. Once verified, the path was transferred to KUKA robot's controller and scaled parts were loaded into the input bin in correct order, and then palletized using the robotic arm. Two cup suction gripper was used as an end effector by the robot to handle the parts. Stack density and stability were qualitatively measured as the parts were

palletized by the robotic arm. The resulting stacks were compared to the simulated trials in Unity as well. Figure 26 shows a representative palletized assembly of two trusses in a robot cell.

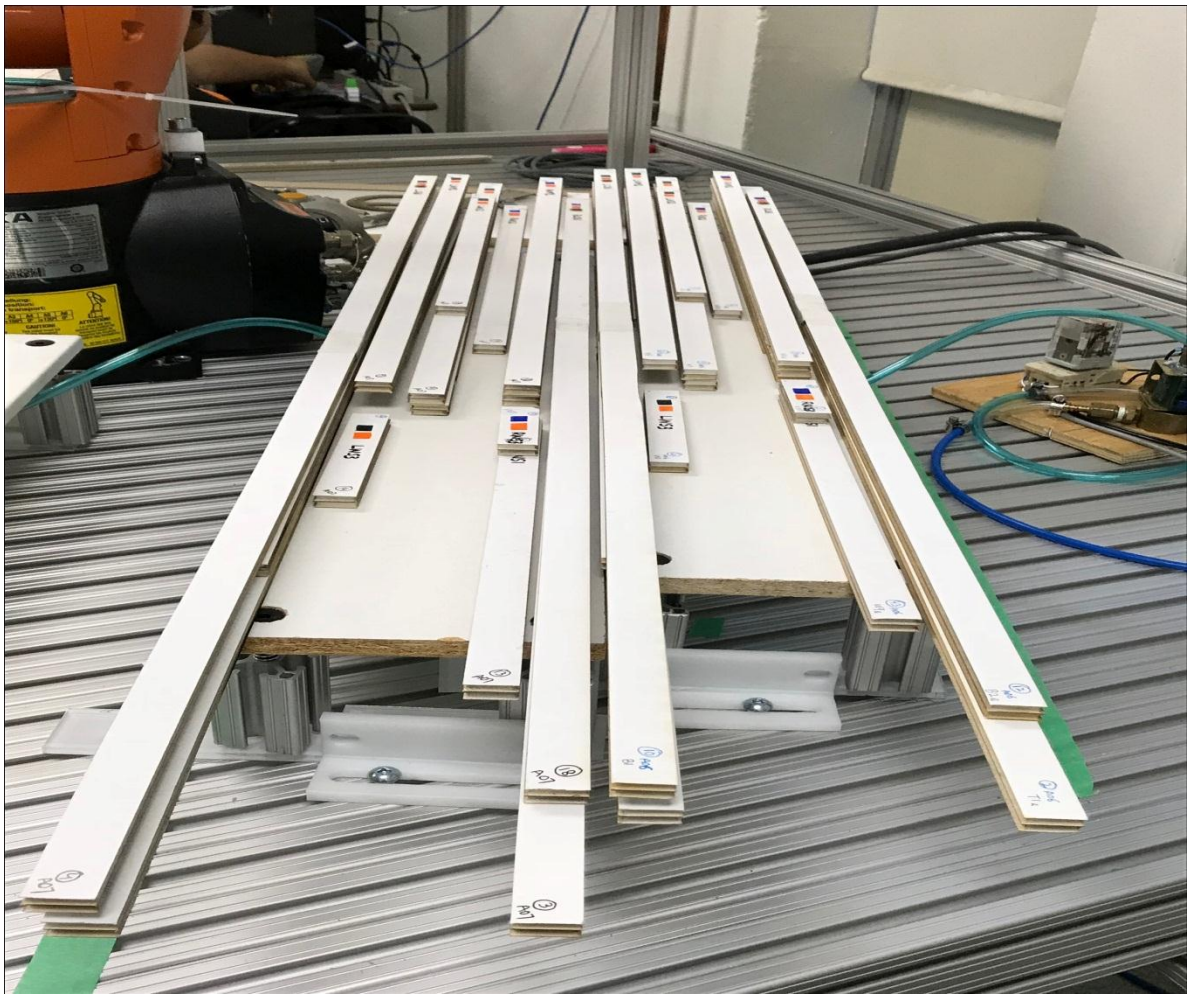


Figure 26: Stacking Assembly of Trusses

Various saw files were simulated in Unity to verify the bin packing method. Quantitative data was generated to further analyze the effectiveness of the process and graphs were produced based on volume utilization and packing density metrics. The following chapter provides details about the results from the implementation in Unity and robot cell.

5. Results

Experimental results of bin packing simulation and physical testing using the robotic arm are presented in this chapter. Saw files provided by the local truss manufacturer were used in Unity simulations while representative stacked truss assemblies in physical testing were compared with their simulation counterparts. By considering the experimental results of simulations in Unity, stability threshold was set to 10% for all the graphs as parts stacking showed optimal results at this value without limiting the potential placement locations, and placing the parts with full stability.

5.1. Saw Files Simulation Results

As mentioned above, simulations were created in Unity using the saw cut files provided by the local truss manufacturer. Parts in saw files varied from 8 to 20 parts per truss. To better understand the packing process, two categories of saw files were created; one with average of 17 parts per truss and the other with average of 12 parts per truss. Assessment methods described in Chapter 4 were implemented and the results are shown in Figures 27 to 34 for both categories of saw files separately. Overall volume utilization and in-bin volume utilization metrics measure the volume utilized by the parts with overhang and with-out overhang respectively. Similarly, overall bin packing density and in-bin packing density metrics respectively shows the packing density of the parts with overhang and without overhang.

5.1.1. Saw Files with Average of Seventeen Parts per Truss

First, tests were performed for packing volumes and densities for the saw files with average of 17 parts per truss. Packing volume and packing density tests were used to validate the bin packing method and to provide the quantitative data on the stacking efficiency of the method.

The saw files are numbered from 1 to 8 with their respective number of parts and assessment results as shown in the table below and summarized in Figures 27 to 30.

Table 2: Summarized Simulation Results for Stacking Efficiencies for Average 17 Parts per Truss

Saw file	No. of Parts	Left Vol	Left in-bin Vol	Left in-bin density	Left density	Right Vol	Right in-bin Vol	Right in-bin density	Right density
1	20	83.95%	56.65%	68.58%	61.11%	58.51%	38.23%	46.28%	42.59%
2	20	83.24%	56.65%	68.58%	60.59%	59.35%	38.79%	46.96%	43.2%
3	18	77.16%	53.29%	64.5%	56.17%	59.35%	41.35%	50.06%	46.54%
4	17	80.55%	51.05%	61.8%	58.64%	56.81%	40.82%	56%	54.2%
5	17	78.01%	52.73%	63.84%	56.79%	58.51%	39.35%	47.63%	42.59%
6	17	77.16%	51.05%	61.8%	50.55%	52.57%	37.67%	45.06%	41.22%
7	17	66.28%	48.82%	59.1%	48.25%	54.41%	42.23%	51.13%	42.66%
8	16	67.2%	47%	46.8%	32.5%	53%	40.6%	43.52%	33.37%

This data is summarized in the figurers below:

Overall volume filled by the parts is shown in Figure 27, where blue and red lines show the volume filled by left and right side parts respectively. This metric compares the left or right side volume of the parts including overhang against the volume of the bin for either side. The graph shows general trend of the volume filled by the truss parts in a bin which varies depending on the number of parts in a truss as well as the truss composition. Green line shows the volume filled in the whole bin.

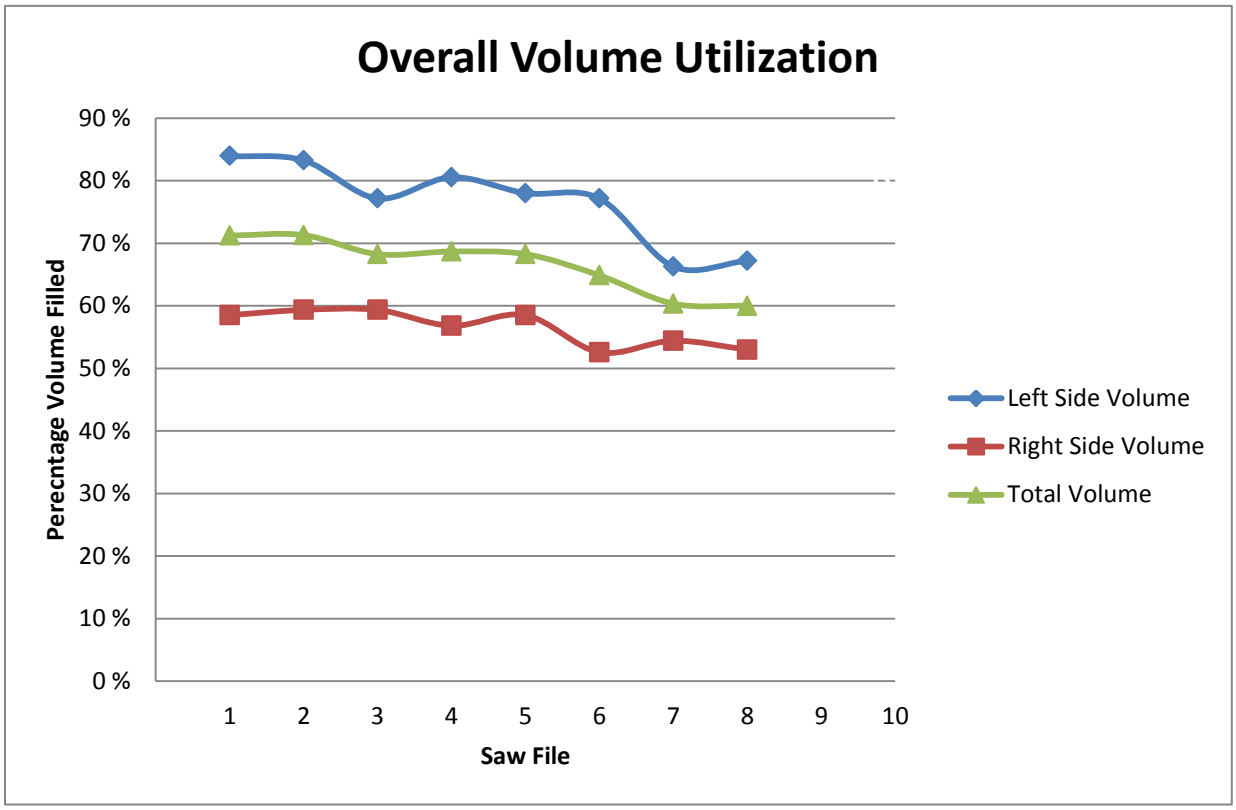


Figure 27: Overall Volume Utilization

In-bin volume utilization metric only considers the parts within the bin boundaries while calculating the packing volume. The resulting efficiencies are lower than those above as part's

overhang is not considered. Left and right side as well as total in-bin volume utilization is shown in the figure below.

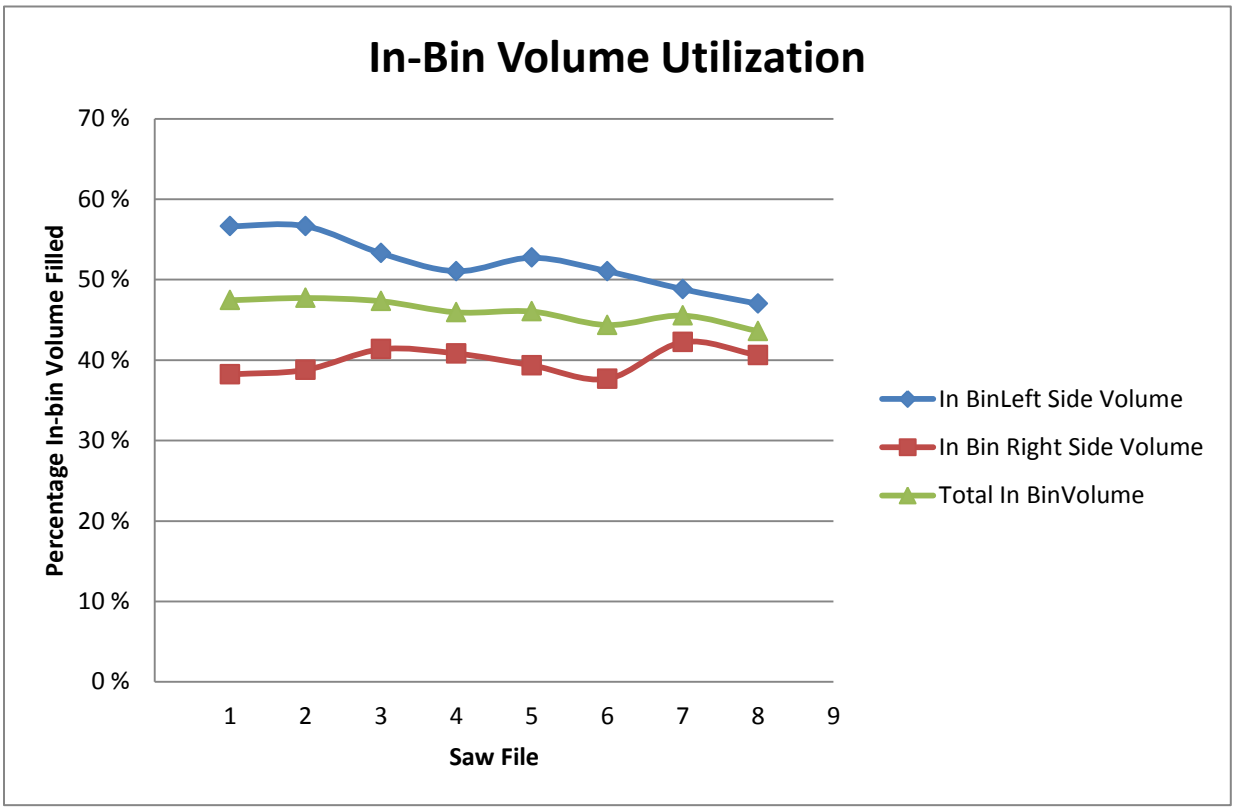


Figure 28: In-Bin Volume Utilization

Parts volume may be less as compared to the bin volume and hence volume utilization metrics might not reflect the proper usage of the bin. As such, packing density metrics were used to measure the volume of the parts by considering the smallest bounding box that fits all the parts instead of bin volume. Overall packing density metric measures the parts packing density including the overhang. Left and right packing densities are shown separately in the figure below.

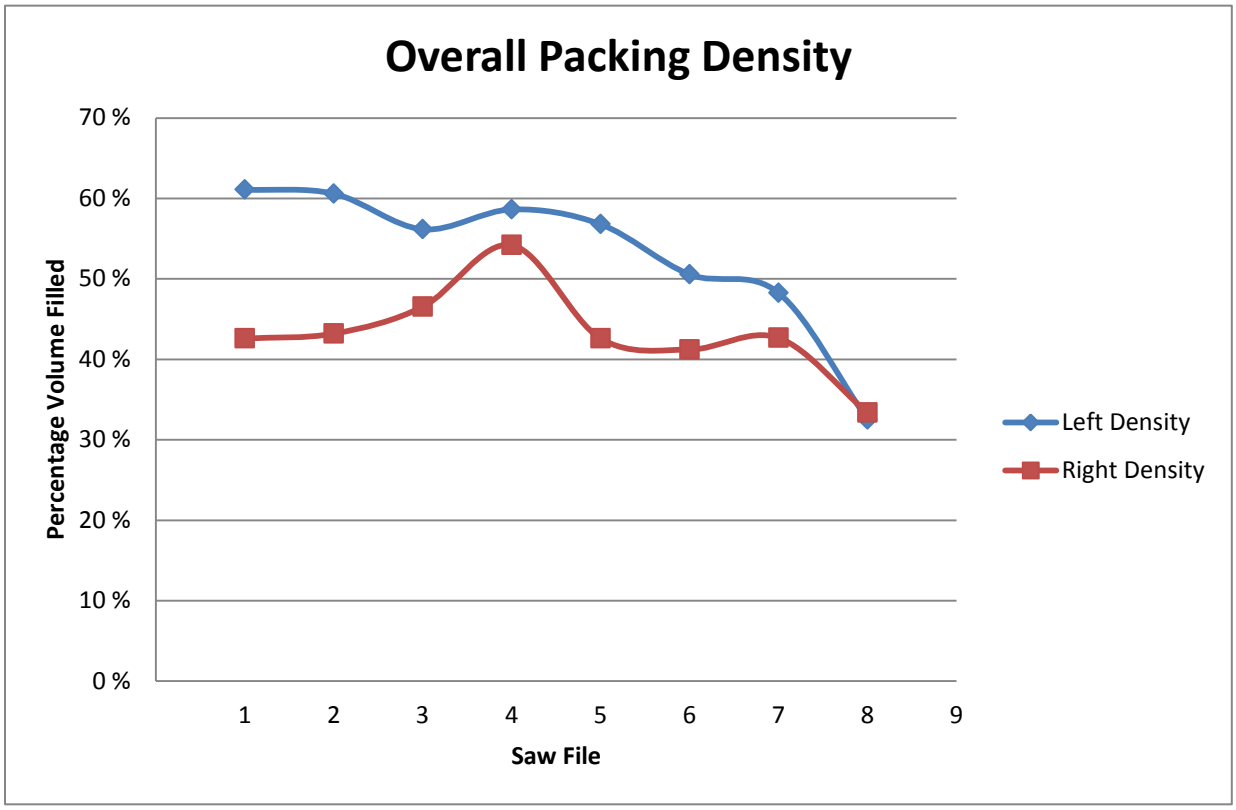


Figure 29: Overall Packing Density Utilization

In-bin packing density of the parts is also used to measure the volume utilization by restricting the parts volumes within the bin boundaries and neglecting the overhanging areas of the parts. The results are shown to have higher in-bin packing densities as compared to the overall packing densities.

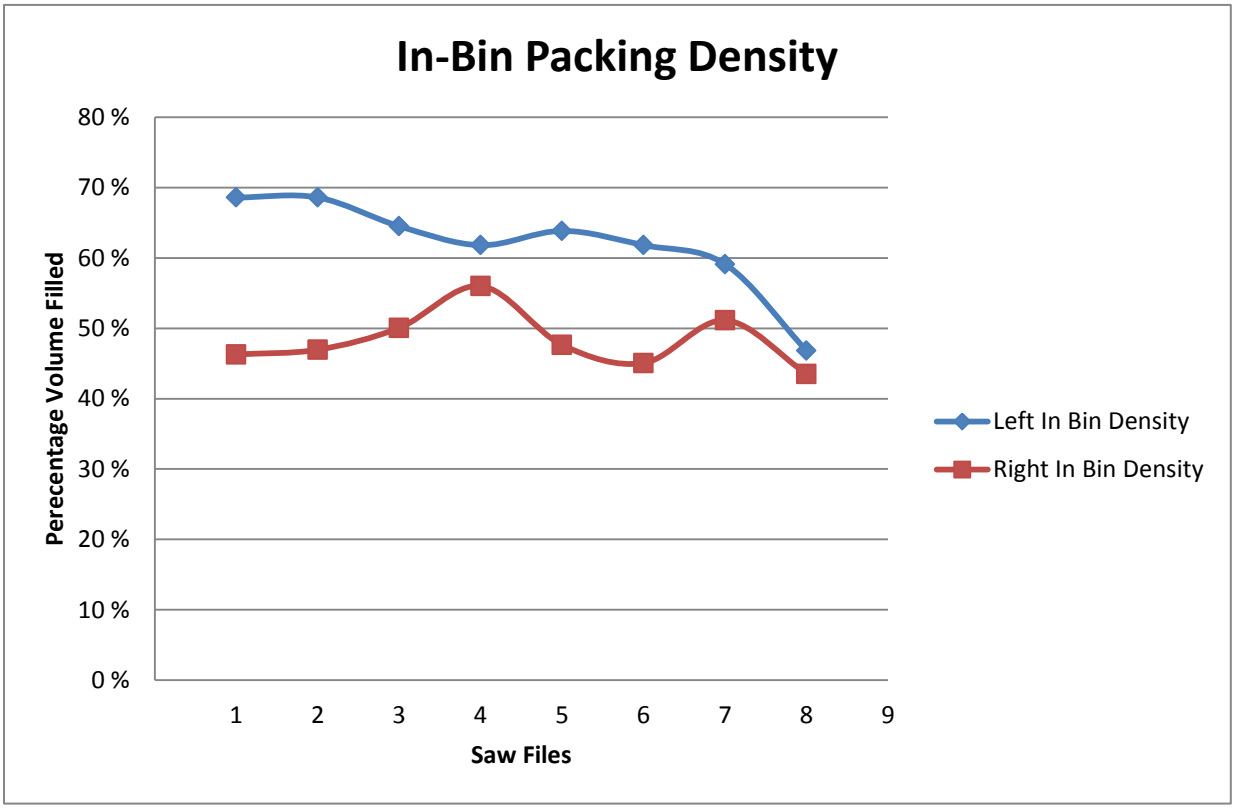


Figure 30: In-Bin Packing Density Utilization

5.1.2. Saw Files with Average of Twelve Parts per Truss

Similarly, tests were performed with the set of saw files with average of twelve parts per truss. The saw files are numbered from 1 to 16 with their respective number of parts and assessment results as shown in the Table 3. The results of these tests shown in the table below include packing volume and packing density utilizations for the left and right side parts for each saw file. The summary of these tests is shown in the Figures 31 to 34 that present the general trend in volume utilization.

Table 3: Summarized Simulation Results for Stacking Efficiencies for Average 12 Parts per Truss

Saw File	No of parts	Left Vol	Left in-bin Vol	Left in bin density	Left density	Right Vol	Right in-bin Vol	Right in-bin density	Right density
1	12	36.46%	29.31%	53.22%	48.67%	34.76%	28.61%	51.95%	46.41%
2	12	34.76%	28.61%	51.95%	46.41%	36.46%	29.31%	53.22%	48.67%
3	13	35.19%	29.03%	52.72%	46.98%	36.46%	29.31%	53.22%	48.67%
4	10	39.855%	28.86%	52.42%	43.51%	30.52%	24.32%	44.17%	33.88%
5	14	50.03%	40.06%	72.75%	54.62%	25.43%	21.55%	39.14%	31.06%
6	14	47.06%	35.8%	65%	57.62%	23.31%	19.71%	35.8%	34.04%
7	14	46.63%	39.62%	71.94%	63.69%	33.07%	29.61%	53.76%	52.45%
8	12	44.09%	30.01%	47.22%	39.27%	50.87%	45.29%	82.24%	76.55%
9	10	45.79%	39.58%	71.88%	64.28%	27.98%	23.51%	42.69%	27.91%
10	10	45.79%	39.58%	71.88%	64.28%	20.35%	15.35%	27.87%	27%
11	9	45.79%	39.5%	71.88%	64.28%	20.35%	15.35%	27.87%	27%
12	10	50.03%	38.5%	69.9%	49.1%	23.31%	17.85%	32.41%	34.22%
13	10	45.79%	33.55%	60.92%	45%	22.89%	18.8%	34.14%	34.45%
14	11	43.24%	39.15%	71.1%	65.05%	24.16%	22.23%	40.38%	26.82%
15	11	23.31%	20.3%	36.86%	33.28%	10.17%	8.73%	15.85%	17.16%
16	8	25.86%	22.41%	40.7%	33.28%	12.17%	10.41%	18.9%	20.86%

Based on the data shown in Table 3, volume utilization with and without overhang along with the packing densities of left and right side parts are summarized in the figures below. Parts in saw files ranged from 8 to 14 parts per truss. Volume utilization and packing density varied depending on the number of parts and overall composition of a truss.

Figures 31 and 32 show the overall volume utilization and in-bin volume utilization of the truss parts where the blue line shows left side parts volume and the red line shows right side parts volume. The resulting efficiencies of in-bin volume utilization are lower than those of overall volume utilization as part's overhang is not considered.

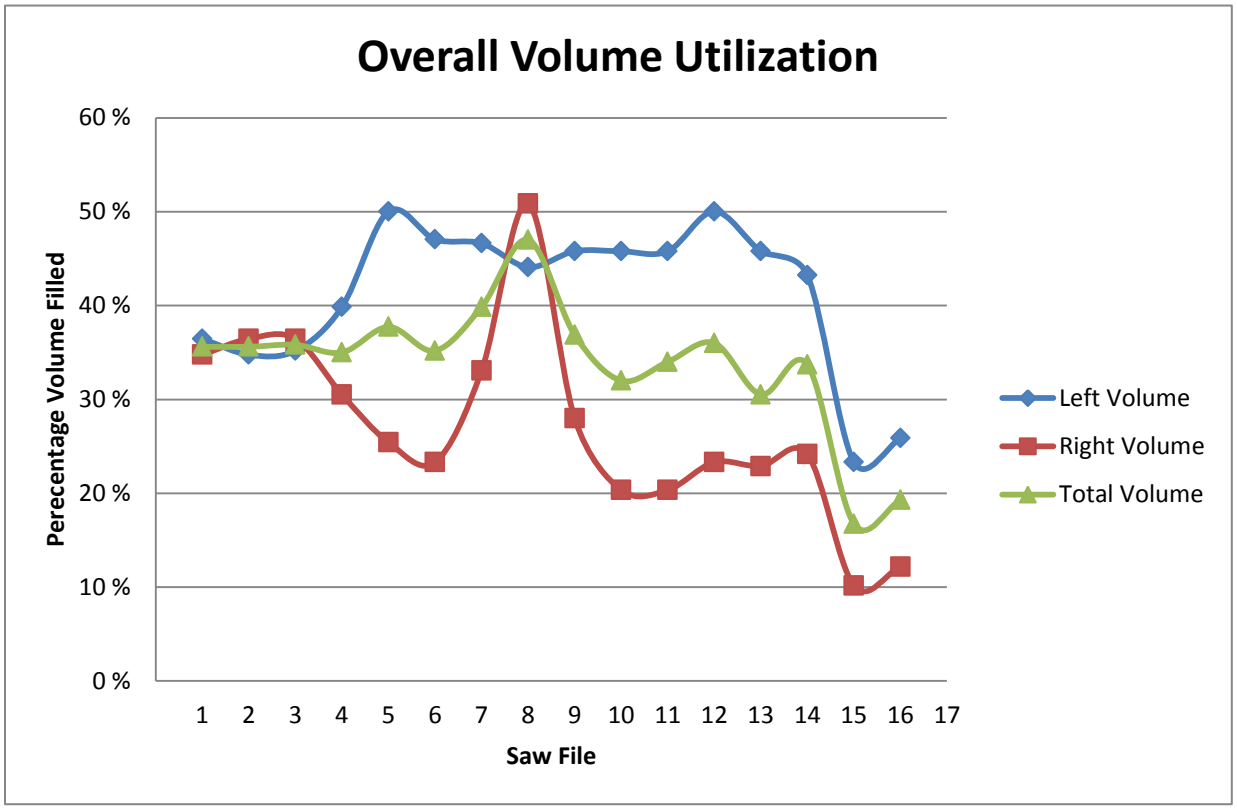


Figure 31: Overall Volume Utilization

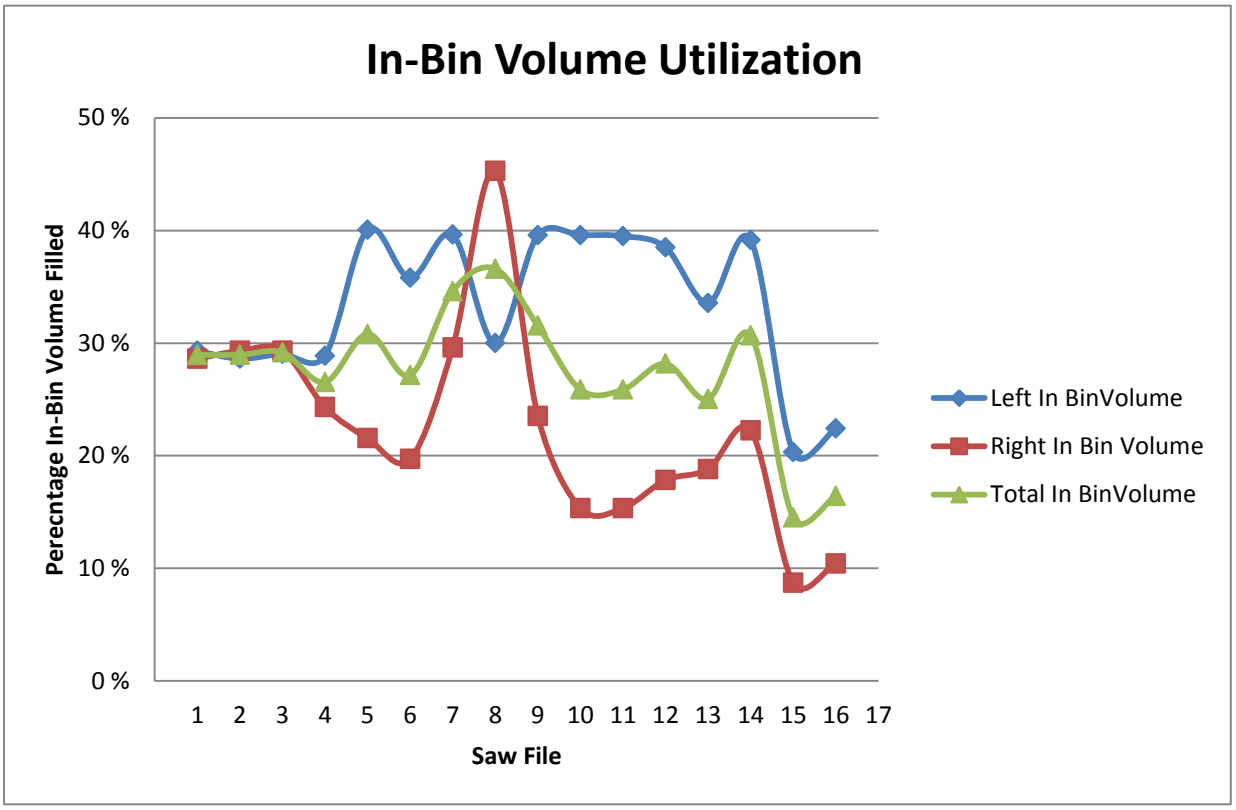


Figure 32: In-Bin Volume Utilization

Figures 33 and 34 show the overall packing density and in-bin packing density respectively for the saw files 1 to 16 as mentioned in Table 3. Left and right side parts packing densities are respectively shown in blue and red lines. The in-bin packing density results are shown to have higher packing densities as compared to the overall packing densities.

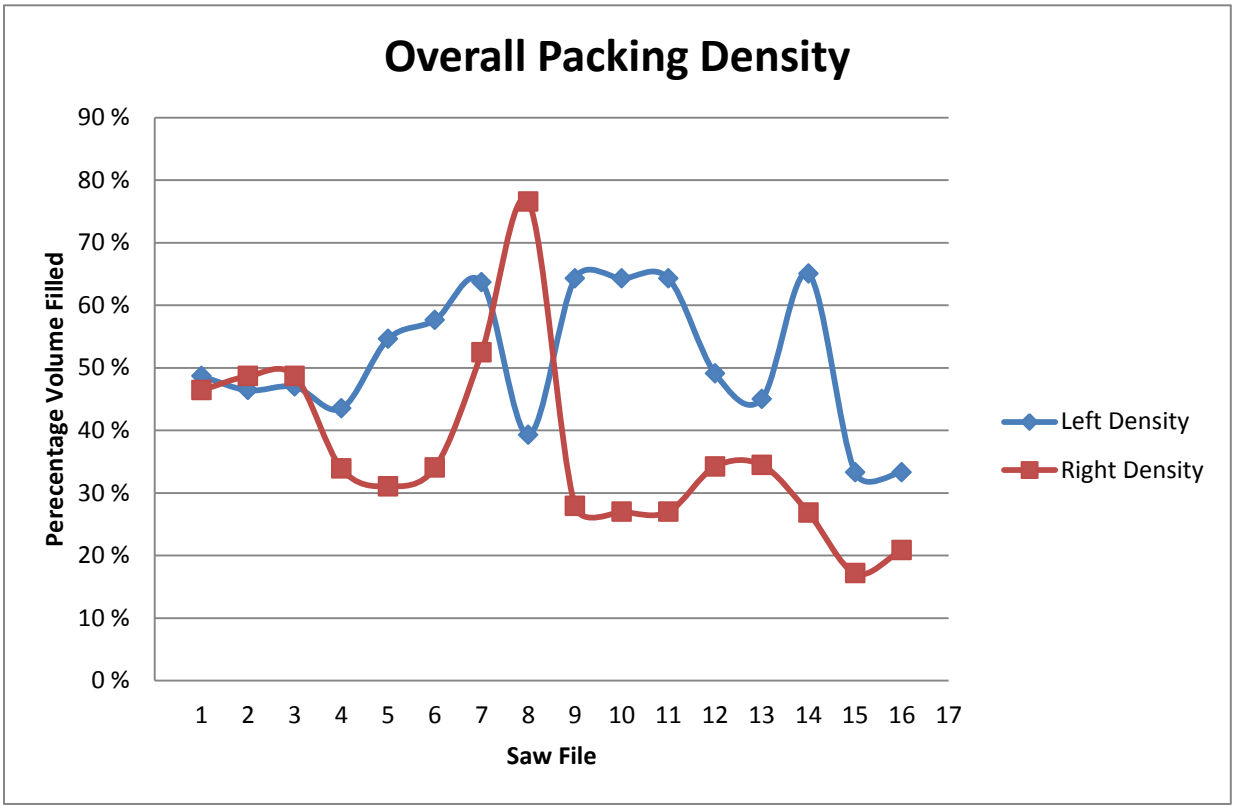


Figure 33: Overall Density Utilization

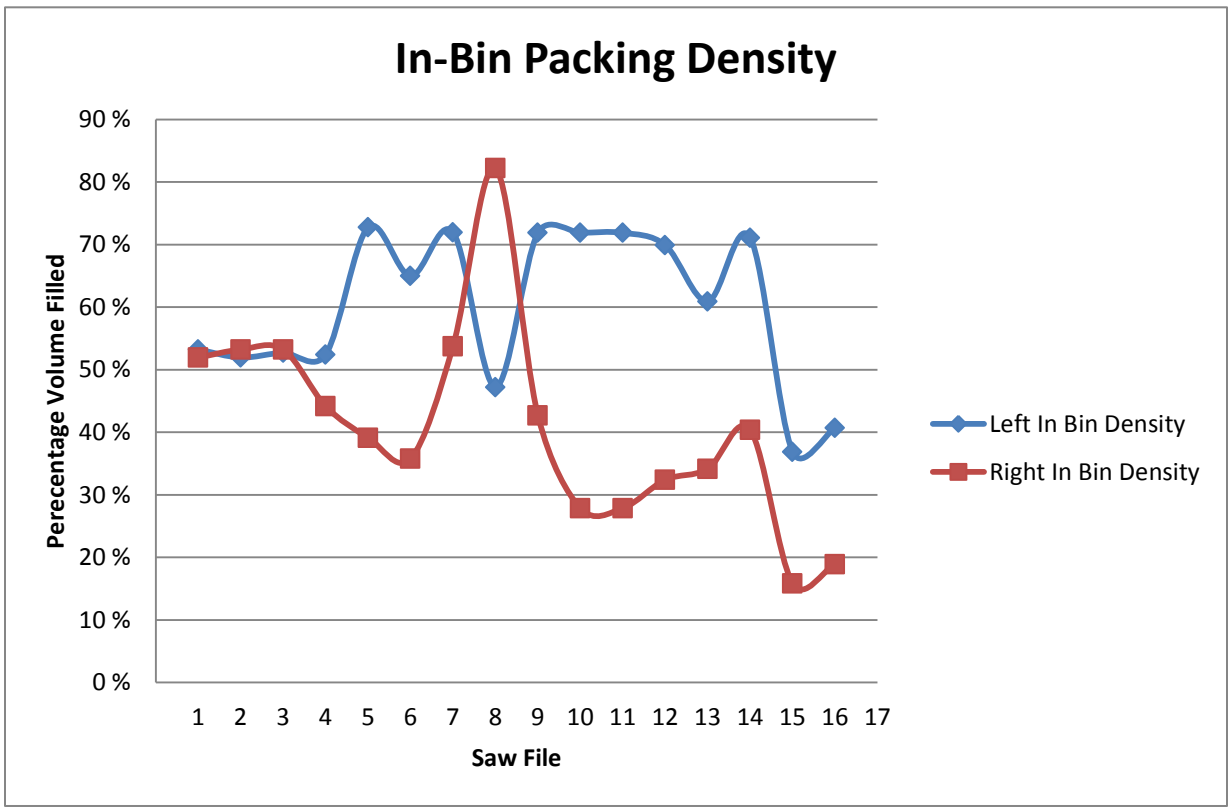


Figure 34: In-Bin Density Utilization

5.2. Results from Tests Conducted using the Experimental Cell

Scaled robotic cell was used to test the real time performance of the method after completing the simulation trails using the robotic arm to ensure that system's path planning, collision detection and stacking algorithms performed as intended. Figure 35 shows the complete assembly of four trusses, stacked into the bins.

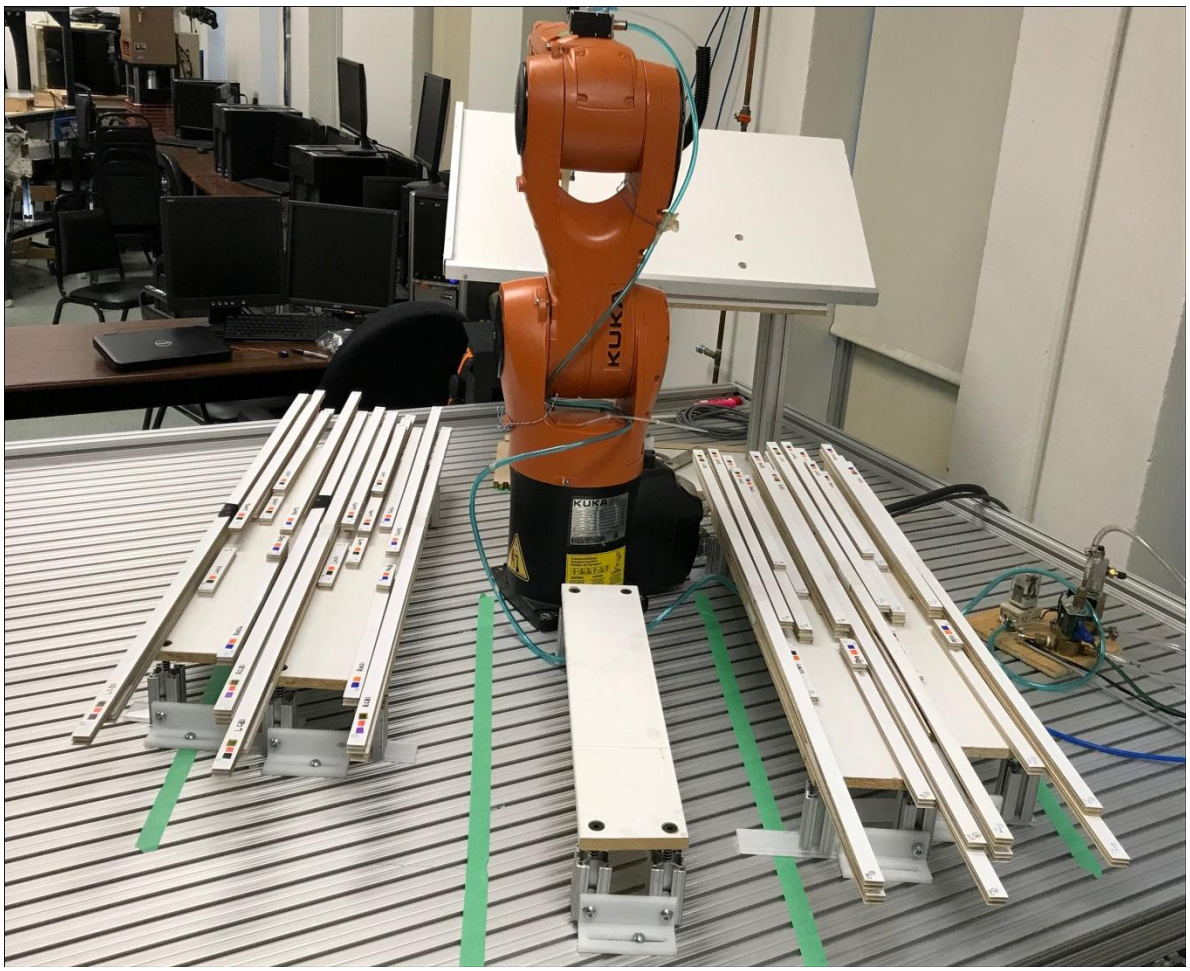


Figure 35: Stacking Assembly of Four Trusses

Four different trusses were used in physical testing based on the manufacturer provided saw files, varying in length from 6 inches to 18 feet and scaled to 1:4.4 for robotic cell. Robot was operated at 75% of its speed to avoid excessive vibrations from the rapid movements. The vibration induced can be eliminated or minimized by mounting the robot to a concrete floor as would be the case in the industry. The experimental robot in the laboratory was mounted on top of a table and hence complete elimination of vibration could not be avoided. Average packing time was determined to be 11 seconds per part using robotic arm. During physical testing, minimum shifting of the parts was observed with full part stability. Simulation trials were compared to the

physical testing results, and both matched as well as system's path planning and packing algorithms performed as required. Figure below compares the simulation of four trusses with physical testing using robotic arm.

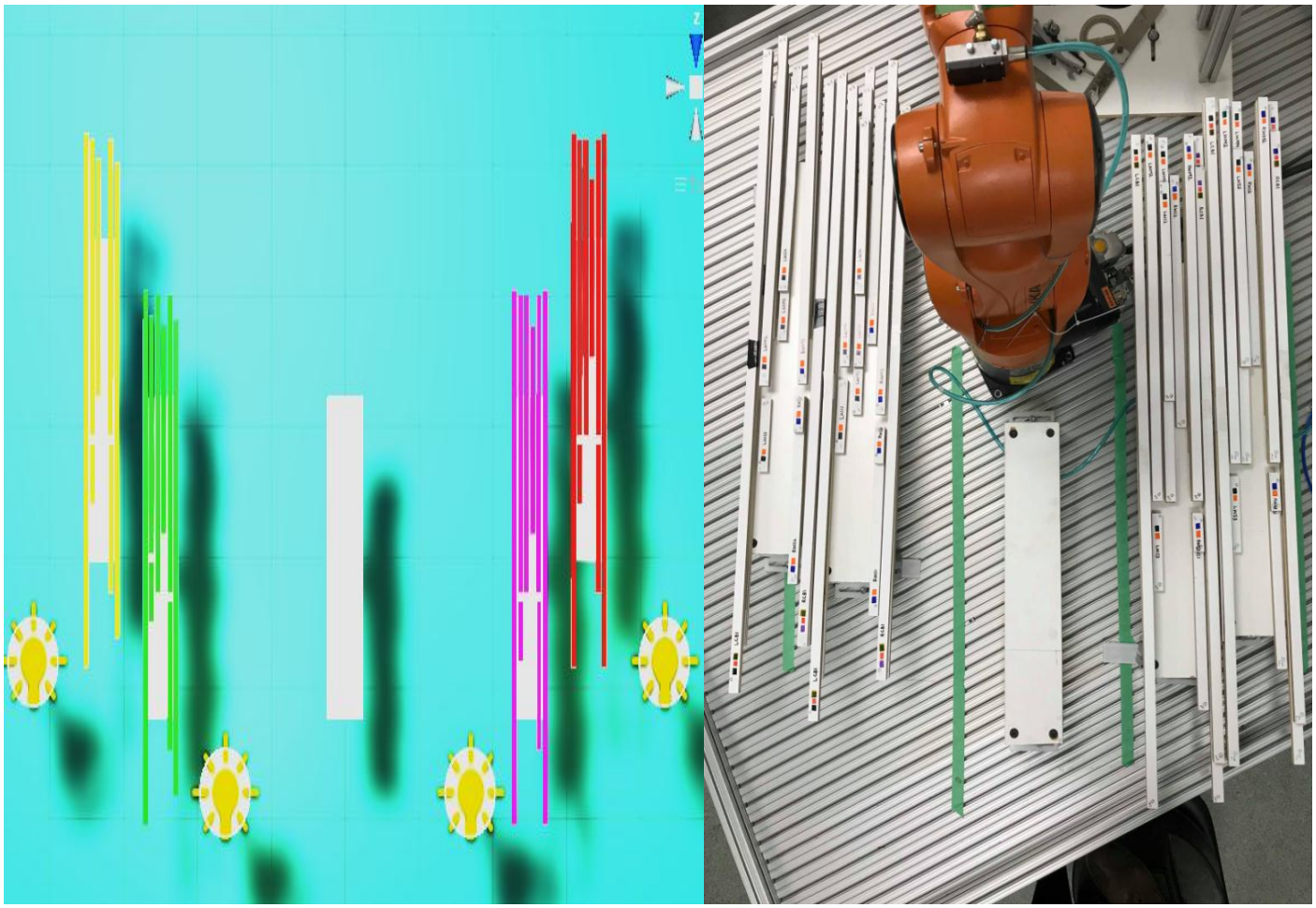


Figure 36: Simulation (Left) Vs. Physical Testing (Right)

To verify the truss assembly from individual parts, trusses were assembled from the parts stacked into the bins. The sequence of stacking allowed assembling the truss with ease. Assembled trusses matched with the design sheets provided by the manufacturer as shown in Figures 37 and 38.

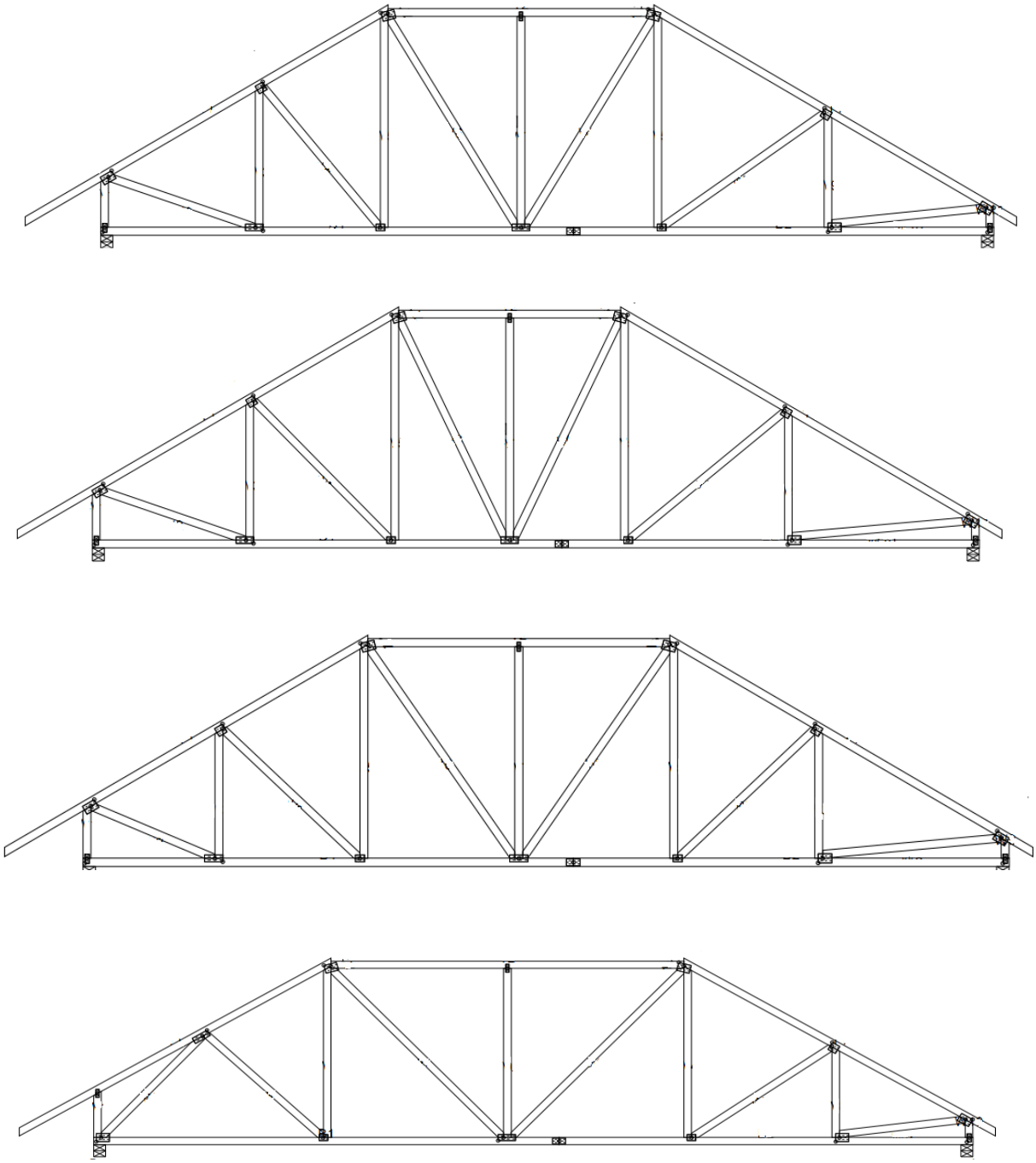


Figure 37: Digital Design of Trusses

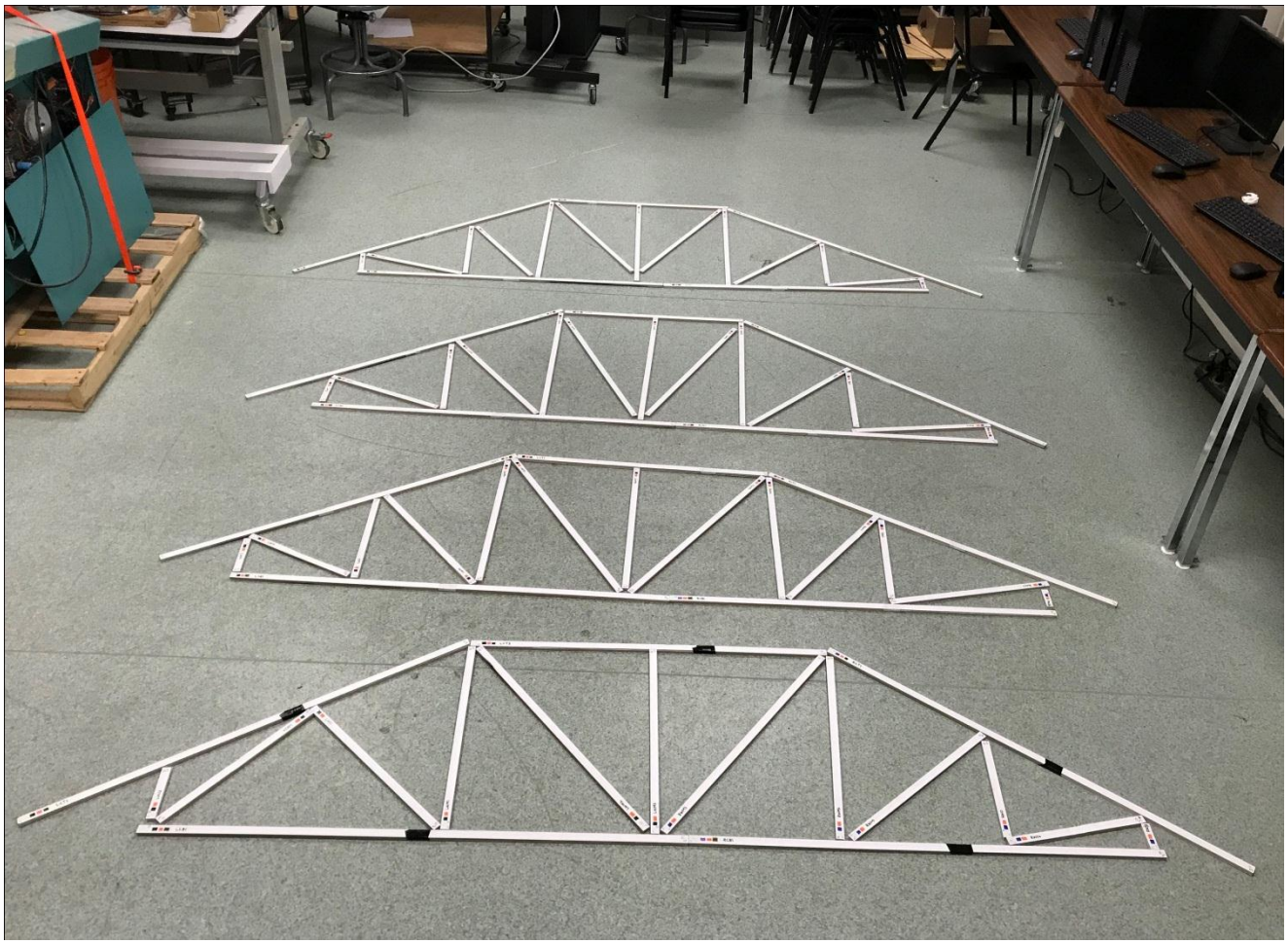


Figure 38: Assembly of Trusses from Stacked Parts

6. Discussion

6.1. Overall System Performance

The overall system performance was very good; successfully stacking the parts by overhanging them beyond the bin boundaries with full stability. The cumulative computation time for a batch of four trusses having 72 parts was 2.99 minutes. The calculation time increases as the number of parts increases as shown in the figure below. All the computations were performed on a 3.40GHz i7-2600 processor.

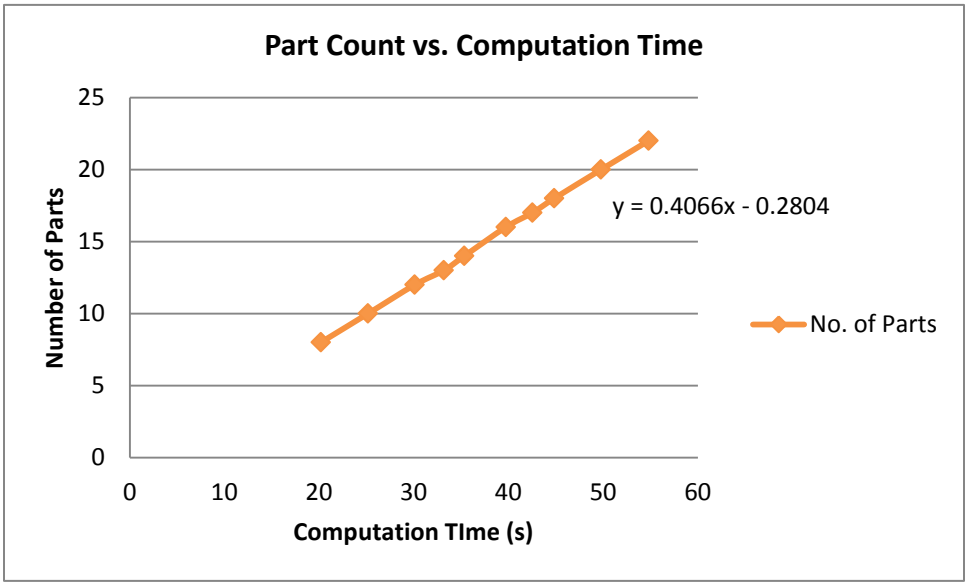


Figure 39: Part Count vs. Computation Time

Figure 27 shows bin volume utilization. It also shows that average bin space utilization was more than 65% for a 17 part truss while considering overhang. Bin densities are lower than those of the volume utilization as shown in Figures 29 and 30. Such lower densities are attributed to the large bounding box generated by the longer parts than the bin itself.

6.2. Comparison

This methodology was developed in two stages. In the initial stage that will be referred to as phase I [1], primary solution was developed and implemented using robotic arm while in second stage, phase II, methodology was further developed and implemented as described in the previous chapters. Phase I & II, both used Unity as a simulation environment and same robotic arm for implementation on the robot cell. Buffer bin was also used in both phases for temporary storage of parts. Following is the detailed comparison about improvement/enhancement of the project to make it more efficient.

6.2.1. Simulation

Unity was chosen to be the simulation environment for the development of the methodology in both phases but in phase II, the simulation setup was improved to mimic the factory like arrangement where an input bin was added along with the sorting bins and the buffer bin. Gravitational and frictional affects were added to the input bin to show actual simulation of the incoming parts from the saw. Part by part stacking could be clearly visualized in simulation and qualitative analysis of the packing methodology could be observed as process continues. Gravitational affects were also added to the parts to observe the part's stability and collision in simulation.

Visual beacons were also added alongside each bin, which provides a visual feedback by illuminating when all the parts of a truss in a particular bin are stacked. Also, visual display was added to the simulation screen that showed the progress of the stacking process as well as the status of packing completion. Time of completion of simulation was also shown on the display.

6.2.2. Methodology

The methodology and packing strategy was greatly emphasized in phase II to improve the packing process and make it easier for the assembler to collect the stacked parts from the bin to decrease the assembly time and increase the production rate. In phase I [1], parts were stacked in the bins without further sorting. In a factory setup, two workers are employed to assemble the parts into a truss. One of them assembles the left side parts while the other assembles the right side parts and both side parts are joined together into a truss. In phase II, when parts appeared in the input bin, they were further sorted for placement in the bins. First, it was checked either the part belonged to left or right side of the truss. Then it was further sorted either as either it is a chord or non-chord part. The part is then placed in the sorting bin according to the methodology as described in Chapter 3. Without sorting the parts into left and right sides, the assembler will have to find the required part by inspecting the complete stack in the bin. This would result in additional labor cost, time and effort during assembly. Phase II developed the stacking methodology to eliminate the time wastage that can significantly reduce time for assembly and increase the production rate.

6.2.3. Path Planning of the Robot

Unity generates the path file for ‘Octopuz’, which is used as a path planner for the industrial robots. In phase I [1], robot’s path had some redundant movements that resulted in longer time for placing a part in the bin. Such movements were eliminated in phase II by minimizing the linear movements of the robot, placing the parts closer to the bin surface to reduce the impact of drop, and reducing the readjustment movements of the end effector. It was recorded in the experiments that for the same truss the average time for placing a part was reduced by 30%. It must be noted that ‘Octopuz’ optimizes the path of robot between the points using built in features. The table

below summarizes the comparison between the work that preceded this work [1] and the work undertaken in this research.

Table 4: Comparison between Previous and Current Work

	Previous Work (Phase-I) [1]	Current Work (Phase-II)
Stack simulation and implementation on the robot cell	Yes	Yes
Use of buffer bin as an intermediate storage	Yes	Yes
Development of factory like setup with input bin and beacons in Unity	No	Yes
Visual Display of the packing process in Unity	No	Yes
Parts' stacking into left and right sides	No	Yes
Parts sorting in chord and Non-Chord parts with further sorting into middle and side parts	No	Yes
Octopuz path optimized	No	Yes
Decrease in average stacking time per part	-	30%

7. Conclusion and Recommendations

7.1. Conclusion

The palletization methodology described in this research is capable of placing the parts in the bins, including the parts larger than bin boundaries, repeatedly with full stability. The computation time of 44 seconds for an 18 part truss is acceptable while allowing the part overhang beyond the bin dimensions significantly. The performance of system both in simulation and implementation using the robotic arm has produced expected results. The system's capability of accommodating the part's overhang may be useful to other bin packing applications where overhanging of the parts is acceptable.

Based on the literature searched, this research is the first attempt to place the parts in the bins with further sub-sorting of the parts to help later assembly along with the overhang beyond the bin boundaries. Sorting into left and right side parts as well as chord and non-chord parts allows the assembler to assemble the parts quickly. Implementation of the developed solution in the laboratory using robotic arm has produced the results equivalent to its simulation counterpart. This thesis provides a complete automation solution to a local truss manufacturer for the packing of truss parts coming out of the saw to the otherwise manual packing of the parts in the bins.

Overall, the system has performed and achieved the objectives as described in the problem statement; developing a bin packing method to place the parts in the bins with full stability while allowing overhang as well as help ease the later assembly that allows the workers to assemble the parts quickly from the dense stack, and verification of the methodology in simulation and physical testing prior to being implemented. This research has demonstrated that part's overhang can provide effective solution to bin packing problems. In a real application all that is required is to

analyze the production on a particular setup using the developed methodology and generate robots path file automatically before execution of the task. The programming skills required to program thousands of paths to accomplish the task otherwise will be very minimal. The methodology developed can be easily scaled up to any robot small or large with ease.

7.2. Recommendations

Various areas are being identified for future work where improvements can be made to make the developed solution more effective. The algorithm developed in this research is greedy and based on the heuristic approach but there is room for implementing other bin packing strategies to check the performance of the system. Different genetic algorithms as well as other bin packing methods such as 'Simulated Annealing' and 'Guillotine Cut' can also be applied and their effectiveness can be compared. Also, greedy system is considered for packing the parts but in future, non-greedy algorithm may be implemented to improve the volume utilization and overall packing efficiency.

As the parts arrive at the gravitational bin, parts are unlikely to arrive in a very organized orientation. The system is open loop at the gravity bin. Commercial vision systems are available that would detect the orientation of the parts and automatically adjust the robot's orientation. This could not be implemented due to unavailability of funds to purchase the system. The complete vision package will be close to $20,000 to purchase and will be an add-on tool to the existing software. The experimental cell relies on external mechanisms to make sure that parts are in correct orientation in the input bin. It is highly recommended to implement vision system or other sensors to make a closed loop system. By the inclusion of sensors, a system can be developed that would allow the packing process by adapting the path planning according to the parts' arrival orientation in the gravity bin.

Other applications where bin packing technique with overhang can be implemented is also an additional area to explore that allows extending the oversized parts beyond the bin boundaries and performance of such a system can be compared to analyze the improvements, if any.

8. References

[1] Matthew Driedger, "Palletization Method for Oversized Part Stacking with an Industrial Robotic Arm," M.Sc. Thesis, University of Manitoba, Winnipeg, 2018.

[2] Silvano Martello, David Pisinger, Daniele Vigo and Edgar den Boef, "Algorithm 864: General and robot-packable variants of the three-dimensional bin packing problem," *ACM Transactions on Mathematical Software*, vol. 33, no. 1, 2007.

[3] Daniel Mack and Andreas Bortfeldt, "A heuristic for solving large bin packing problems in two and three dimensions," *Central European Journal of Operations Research*, vol. 20, no. 2, p. 337–354, 2012.

[4] Silvano Martello and Michele Monaci, "Models and algorithms for packing rectangles into the smallest square," *Computers and Operations Research*, vol. 63, pp. 161-171, 2015.

[5] S. J. You and S. H. Ji, "Design of a multi-robot bin packing system in an automatic warehouse," in *1th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, Vienna, Austria, 2014.

[6] Abderrahmane Aggoun, Ahmed Rhiat and François Fages, "Panorama of real-life applications in logistics embedding bin packing optimization algorithms, robotics and cloud computing technologies," in *3rd IEEE International Conference on Logistics Operations Management*, Fez, Morocco, 2016.

[7] Stephen Balakirsky, Thomas Kramer and Frederick Proctor, "Metrics for Mixed Pallet Stacking," in *Performance Metrics for Intelligent Systems (PerMIS) Workshop*, Baltimore, USA, 2011.

[8] S. Vargas-Osorio and C. Zuniga, "A Literature Review on the Pallet Loading Problem," *Estilo de Citación de Artículo*, no. 15, pp. 69 - 80, 2016.

[9] Tamas Bartok and Csanad Imreh, "Heuristic algorithms for the weight constrained 3-dimensional bin packing model," in *IEEE International Symposium on Logistics and Industrial Informatics*, Smolenice, Slovakia, 2012.

[10] F. Luo, C. H. Gu and S. L. Huang, "A fast sequential bin packing algorithm with predefined fill level," in *2015 International Conference on Smart and Sustainable City and Big Data (ICSSC)*, Shanghai, China, 2015.

[11] Yi-Ping Cui, Yaodong Cui and Tianbing Tang, "Sequential heuristic for the two-dimensional bin-packing problem," *European Journal of Operational Research*, vol. 240, no. 1, pp. 43-53, 2015.

[12] K. Fleszar, "Three Insertion Heuristics and a Justification Improvement Heuristic for Two-

dimensional Bin Packing with Guillotine Cuts," *Computers & Operations Research,* vol. 10, no. 1, pp. 463-474, 2012.

[13] S. Hong, D. Zhang, H. C. Lau, X. Zeng and Y. W. Si, "A Hybrid Heuristic Algorithm for the 2D Variable-Sized Bin Packing Problem," *European Journal of Operational Research,* vol. 238, no. 1, pp. 95-103, 2014.

[14] Roberto Aringhieri, Davide Duma, Andrea Grosso and Pierre Hosteins, "Simple but effective heuristics for the 2-constraint bin packing problem," *Journal of Heuristics,* vol. 24, no. 3, p. 345–357, 2018.

[15] Jens Egeblad and David Pisinger, "Heuristic approaches for the two- and three-dimensional knapsack packing problems," *Computers & Operations Research,* vol. 36, no. 4, pp. 1026-1049, 2009.

[16] E. Lopez-Camacho, H. Terashima-Marin, P. Ross and G. Ochoa, "A Unified Hyper-Heuristic Framework for Solving Bin Packing Problems," *Expert Systems with Applications,* vol. 41, no. 15, pp. 6876-6889, 2014.

[17] Geunsoneg Jung, Xu Jing, Jaehyuk Cha and Sungjae Han, "Enhancing the assessment item rearrangement with bin packing algorithms," in *International Conference on Information Science and Security (ICISS)*, Seoul, South Korea, 2016.

[18] Taha Ghasemi and Mohammad Reza Razzazi, "Analysis of a first-fit algorithm for the capacitated unit covering problem," *International Journal of Computer Mathematics,* vol. 94, no. 7, pp. 1375-1389, 2016.

[19] Lei Huang, Zhong Liu and Zhi Liu, "An improved lowest-level best-fit algorithm with memory for the 2D rectangular packing problem," in *International Conference on Information Science, Electronics and Electrical Engineering*, Sapporo, Japan, 2014.

[20] Aristide Grange, Imed Kacem and Sebastien Martin, "Algorithms for the bin packing problem with overlapping items," *Computers & Industrial Engineering,* vol. 115, pp. 331-341, 2018.

[21] G. Stavrinides and H. Karatza, "Scheduling Multiple Task Graphs in Hetrogeneous Distributed Real-Time Systems by Exploiting Schedule Holes with Bin Packing Techniques," *Simulation Modelling Practice and Theory,* vol. 19, no. 1, pp. 540-552, 2011.

[22] Mauro Maria Baldi and Maurizio Bruglieri, "On the generalized bin packing problem," *International Transactions In Operational Research,* vol. 24, no. 3, p. 425–438, 2016.

[23] S. Takahara, "A Multi-start Local Search Approach to the Multiple Container Loading Problem," *Greedy Algorithms, Witold Bednorz (Ed.), InTech,* pp. 55-68, 2008.

[24] Liu Sheng, Zhao Hongxia, Dong Xisong and Cheng Changjian, "A heuristic algorithm for container loading of pallets with infill boxes," *European Journal of Operational Research,* vol. 252, no. 3, pp.

728-736, 2016.

[25] David Pisinger, "Heuristics for the container loading problem," *European Journal of Operational Research,* vol. 141, no. 2, pp. 382-392, 2002.

[26] Hongteng Wu, Stephen C.H. Leung, Yain-whar Si, Defu Zhang and Adi Lin, "Three-stage heuristic algorithm for three-dimensional irregular packing problem," *Applied Mathematical Modelling,* vol. 41, p. 431–444, 2017.

[27] Xiaozhou Zhao, Julia A. Bennell, Tolga Bektas and Kath Dowsland, "A comparative review of 3D container loading algorithms," *International Transactions In Operational Research,* vol. 23, no. 1-2, p. 287–320, 2016.

[28] Batoul Mahvash, Anjali Awasthi and Satyaveer Chauhan, "A column generation-based heuristic for the three dimensional bin packing problem with rotation," *Journal of the Operational Research Society,* vol. 69, no. 1, pp. 78-90, 2017.

[29] Ana Moura and Andreas Bortfeldt, "A two-stage packing problem procedure," *International Transcations in Operational Research,* vol. 24, no. 1-2, pp. 43-58, 2017.

[30] Rommel D.Saraiva, Napoleao Nepomuceno and Placido R.Pinheiro, "A layer-building algorithm for the three-dimensional multiple bin packing problem: a case study in an automotive company," *IFAC-PapersOnLine,* vol. 48, no. 3, pp. 490-495, 2015.

[31] Rasmus R.Amossena and David Pisinger, "Multi-dimensional bin packing problems with guillotine constraints," *Computers & Operations Research,* vol. 37, no. 11, pp. 1999-2006, 2010.

[32] Defu Zhang, Yu Peng and Stephen C.H.Leung, "A heuristic block-loading algorithm based on multi-layer search for the container loading problem," *Computers & Operations Research,* vol. 39, no. 10, pp. 2267-2276, 2012.

[33] H.Gehring and A.ortfeldt, "A genetic algorithm for solving the container loading problem," *International Transactions in Operational Research,* vol. 4, no. 5-6, pp. 401-418, 1997.

[34] C.Joncour, S.Michel, R.Sadykov, D.Sverdlov and F.Vanderbeck, "Column Generation based Primal Heuristics," *Electronic Notes in Discrete Mathematics,* vol. 36, pp. 695-702, 2010.

[35] F. Parreno, R. Alvarez-Valdes, J. M. Tamarit and J. F. Oliveira, "A Maximal-Space Algorithm for the Container Loading Problem," *Informs Journal on Computing,* vol. 20, no. 3, pp. 412-422, 2008.

[36] Silvano Martello, David Pisinger and Daniele Vigo, "The Three-Dimensional Bin Packing Problem," *Operations Research,* vol. 48, no. 2, pp. 189-213, 2000.

[37] Teodor Gabriel Crainic, Guido Perboli and Roberto Tadei, "Extreme Point-Based Heuristics for Three-Dimensional Bin Packing," *Informs Journal on Computing,* vol. 20, no. 3, pp. 368-384, 2008.

[38] Teodor Gabriel Crainic, Guido Perboli and Roberto Tadei, "A two-level tabu search for the three-dimensional bin packing problem," *European Journal of Operational Research,* vol. 195, no. 3, pp. 744-760, 2009.

[39] Minghui Zhang, Yan Lan and Hanxi Li, "A New Bin Packing Algorithm with Buffer," in *International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*, Xiamen, China, 2018.

[40] Shahriar Asta, Ender Ozcan and Andrew J. Parkes, "CHAMP: Creating heuristics via many parameters for online bin packing," *Expert Systems With Applications,* vol. 63, pp. 208-221, 2016.

[41] Mhand Hifi and Labib Yousef, "A hybrid algorithm for packing identical spheres into a container," *Expert Systems With Applications,* vol. 96, pp. 249-260, 2018.

[42] Alessio Trivella and David Pisinger, "The load-balanced multi-dimensional bin-packing problem," *Computers & Operations Research,* vol. 74, pp. 152-164, 2016.

[43] Samir Elhedhli, Fatma Gzara and Yi Feng Yan, "A MIP-based slicing heuristic for three-dimensional bin packing," *Optimization Letters,* vol. 11, no. 8, p. 1547–1563, 2017.

[44] F. Parreno, R. Alvarez-Valdes, J. F. Oliveira and J. M. Tamarit, "Neighborhood structures for the container loading problem: a VNS implementation," *Journal of Heuristics,* vol. 16, no. 1, p. 1–22, 2010.

[45] R.L.Rao and S.S.Iyengar, "Bin-packing by simulated annealing," *Computers & Mathematics with Applications,* vol. 27, no. 5, pp. 71-82, 1994.

[46] Hideichi Nakamoto, Haruna Eto, Takafumi Sonoura, Junya Tanaka and Akihito Ogawa, "High-speed and Compact Depalletizing Robot Capable of Handling Packages Stacked Complicatedly," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, South Korea, 2016.

[47] Shuai D. Han, Nicholas M. Stiffler, Kostas E. Bekris and Jingjin Yu, "Efficient, High-Quality Stack Rearrangement," *IEEE Robotics and Automation Letters,* vol. 3, no. 3, pp. 1608-1615, 2018.

[48] C. Wurll, "Mixed Case Palletizing with Industrial Robots," in *47st International Symposium on Robotics*, Munich, Germany, 2016.

[49] Yunjie Xu, Yumei Liu, Lina Hao and Hongtai Cheng, "Design of Palletizing Algorithm Based on Palletizing Robot Workstation," in *IEEE International Conference on Real-time Computing and Robotics,* Angkor Wat, Cambodia, 2016.

[50] Roshni N and Dr. Sunil Kumar T K, "Pick and Place Robot Using the Centre of Gravity Value of the Moving Object," in *IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS)*, Srivilliputhur, India, 2017.

[51] Erick Dube and Leon R. Kanavathy, "OPTIMIZING THREE-DIMENSIONAL BIN PACKING THROUGH SIMULATION," in *International Conference on Modelling, Simulation and Optimization*, Gaborone, Botswana, 2006.

[52] Navonil Mustafee and Eberhard E. Bischoff, "Analysing trade-offs in container loading: combining load plan construction heuristics with agent-based simulation," *International Transactions in operational Research*, vol. 20, no. 4, pp. 471-491, 2013.

[53] G. Demisse, R. Mihalyi, B. Okal, D. Poudel, J. Schauer and A. Nuchter, "Mixed Palletizing and Task Completion for Virtual Warehouses," in *Virtual Manufacturing and Automation Competition (VMAC) Workshop at the Int. Conference of Robotics and Automation (ICRA)*, 2012.

[54] Stephen Balakirsky and Zeid Kootbally, "Usarsim/Ros: A Combined Framework For Robotic Control And Simulation," in *Proceedings of the ASME 2012 International Symposium On Flexible Automation*, St. Louis, Missouri, USA, 2012.

[55] Martin Schuster, Richard Bormann, Daniela Steidl, Saul Reynolds-Haertle and Mike Stilman, "Stable Stacking for the Distributor's Pallet Packing Problem," in *International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, 2010.

[56] Leonardo Junqueira and Reinaldo Morabito, "On solving three-dimensional open-dimension rectangular packing problems," *Engineering Optimization*, vol. 49, no. 5, pp. 733-745, 2016.

[57] A.Bortfeldt, H.Gehring and D.Mack, "A parallel tabu search algorithm for solving the container loading problem," *Parallel Computing*, vol. 29, no. 5, p. 2003, 641-662.

[58] T. Sorensen, S. Foged, J. M. Gravers, M. N. Janardhanan, P. Nielsen and O. Madsen, "3D Pallet Stacking with Rigorous Vertical Stability," *Distributed Computing and Artificial Intelligence*, vol. 474, pp. 535-544, 2016.

[59] H. Carpenter and W. B. Dowsland, "Practical Considerations of the Pallet-Loading Problem," *The Journal of the Operational Research Society*, vol. 36, no. 6, pp. 489-497, 1985.