An intelligent framework for computer-aided automation and optimization of Light-Frame Building manufacturing process

by

Nabeel Malik

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in
Civil (Cross-Disciplinary)

Department of Civil and Environmental Engineering
University of Alberta

ABSTRACT

The growth of panelized construction in North America's building construction sector has allowed for the emergence of a viable alternative to traditional construction methods. Panelized construction has proven to be a promising method due to its use of off-site manufacturing technology, which provides a higher quality product with reduced site disruptions and a shorter construction cycle. Recent demand for sustainable, green, and energy-efficient construction has allowed for the implementation of innovative processes, with an aim to optimize pre-existing manufacturing techniques. However, among the existing manufacturing processes in panelized construction, there remains avoidable waste resulting from machine cutting operation, and thus a demand for novel processes to increase the productivity of light-frame panel assembly. This research, focusing on the construction of wood and light-gauge steel-framed panels, presents a methodology that aims to improve the utilization of raw stock during the automatic cutting of wood floor components, and presents a framework for transferring shop drawing information to machine-executable commands for automated production of wall and floor frames. By reducing cutting waste through dynamic reallocation of unused material, generating collision-free tool trajectories by intelligently integrating collision detection and avoidance systems, and implementing readability of such information for wall-framing machinery, the wall and floor framing processes in prefabrication construction can be tailored for increased productivity and reduced waste, respectively. Additionally, by applying throughput time reduction methodology, such information can further guide design improvements in existing wall-framing machinery. This thesis also presents real-world case studies to validate the effectiveness and viability of the research within the domain of prefabricated construction.

PREFACE

This thesis is the original work of Nabeel Malik. Two journal papers and one conference paper related to this thesis have been submitted or published and are listed as below with the numbered list representing their respective chapters. The thesis is organized in a paper-based format.

2. **Malik, N**., Ahmad, R., and Al-Hussein, M. Generation of safe tool-paths for automatic manufacturing of light gauge steel panels in residential construction. *Automation in Construction*. (under review)
3. **Malik, N**., Ahmad, R., Chen, Y., Altaf, S., and Al-Hussein, M. Minimizing joist cutting waste through dynamic waste allocation in panelized floor manufacturing. *International Journal of Construction Management*. (under review)
4. **Malik, N.**, Ahmad, R., and Al-Hussein, M. Generation of optimize tool-paths for automatic manufacturing of timber panels in residential construction. *2019 Modular and Offsite Construction Summit* (in preparation)
5. **Malik, N**., Ahmad, R., and Al-Hussein, M. (2018). Investigating effects of reduced technological constraints on cycle time through simulation modelling for automated steel wall framing. *Proceedings, 2018 Modular and Offsite Construction Summit*, Hollywood, FL, USA, Mar. 22–25.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my supervisor, Dr. Mohamed Al-Hussein, and co-supervisor, Dr. Rafiq Ahmad, for the continuous support and their invaluable feedback and encouragement during this research. I could not imagine having better mentors to guide me during this journey.

In addition, I am grateful for the support of all personnel involved in the machine development group. In particular, I would like to thank Anas, Allen, Pablo, Gurjeet, and Shi for their continuous support and feedback during the design and testing phase of prototype machines. I would also like to express my thanks to Jonathan, Sadiq, Claire, Hamida, and other faculty members and staff in the Modular Construction research group. Also, I want to thank the Natural Sciences and Engineering Research Council of Canada for their financial support.

Finally, I must express my deepest gratitude to my parents, Shahnaz and Mohammad Malik, my siblings, and friends for all the love and encouragement over the years.

Table of Contents

ABSTRACT ii

PREFACE iv

ACKNOWLEDGEMENTS v

List of Tables ix

List of Figuresx

LIST OF ABBREVIATIONS xiii

Chapter 1 Introduction 1

1.1 Background and Motivation 1

1.1.1 Dynamic waste allocation 2

1.1.2 Tool-path generation method for automatic manufacturing of light-frame panels: Steel 3

1.1.3 Tool-paths generation for automatic manufacturing of light-frame panels: Timber 4

1.1.4 Improved automated wall framing 5

1.2 Research objectives 6

1.3 Organization of thesis 7

Chapter 2 Dynamic waste allocation 9

2.1 Introduction 9

2.2 Literature review 10

2.3 Waste minimization: Floor Manufacturing 13

2.4 Model: Dynamic Waste Reallocation 16

2.4.1 Initialization 16

2.4.2 Reading Data 18

2.4.3 Pre-Processing 18

2.4.4 Floor joist Cutting 19

2.4.5 Lateral Bracing Cutting 23

2.4.6 Post-Processing 24

2.5 Case Studies 26

3.1.1 Case Study I 27

3.1.2 Case Study II 28

3.1.3 Case Study III 29

3.1.4 Summary of case studies 30
2.6 Conclusion 31
Chapter 3 Tool-path generation method for automatic manufacturing of light-frame panels: Steel 33
3.1 Introduction 33
3.1.1 Background 33
3.1.2 Literature Review 36
3.2 System Framework: Mapping for manufacturing residential walls (MMRW).. 41
3.2.1 Data Input 43
3.2.2 Data Extraction 45
3.2.3 Path Generation (Guided Nearest Neighbour algorithm) 47
3.2.4 Simulation: Integration of schedule (time) information 54
3.2.5 Collision Detection and Avoidance 59
3.2.6 Post-processing 64
3.2.7 GUI Design 66
3.3 Validation results: Case Studies 66
3.4.1 Case Study 1 67
3.4.2 Case Study 2 68
3.4.3 Case Study 3 69
3.4.4 Case Study 4 70
3.4 Discussion 72
3.5 Conclusions 74
Chapter 4 Tool-path generation for automatic manufacturing of light-frame panels: Wood 76
4.1 Introduction 76
4.1.1 Nailing Station 78
4.1.2 Cutting Station 78
4.1.3 Drilling 78
4.2 Wood wall-framing machine motion planning 79
4.2.1 Data Input 80
4.2.2 Data Extraction 81
4.2.3 Path Generation (Multi-spindle Tool-path Generation Algorithm) 81
4.2.4 Simulation: Integration of schedule (time) information 84

4.2.5 Collision Detection and Avoidance 84
4.2.6 Post-processing 84
4.2.7 GUI Design 86
4.3 Conclusion 87
Chapter 5 Improved automated wall framing 88
5.1 Introduction 88
5.2 System Description 90
5.3 Model Construction 92
5.3.1 Scenario 1: Manual Screw Feeding 94
5.3.2 Scenario 2: Auto Screw Feeding System 95
5.3.3 Scenario 3: Design Modification for Frame Clamping 95
5.3.4 Scenario 4: Hybrid Design with Clamping and feeding 96
5.4 Model Input Data 96
5.5 Results and Discussion 96
5.6 Conclusion 99
Chapter 6 Conclusion 101
6.1 General conclusion 101
6.2 Research contributions 103
6.3 Research limitations 103
6.4 Future research 104
References 106

List of Tables

Table 2.1 Sample summary table for optimizing floor joist cutting (AS = allocated stock, mm) 23
Table 2.2. Summary of cases produced throughout the optimization process........ 24
Table 3.1 Motor parameters used for generating recipe files and for operating steel framing machine (*160.45 Encoder = 1 mm) 67
Table 3.2 Summary of case studies, including a breakdown of screw-fastening operations and cycle time analysis 74
Table 4.1 Wood Framing Machine—Operational Categories 85
Table 5.1 Summary of select variables from simulation models (Local = L and Global = G). 93
Table 5.2 Summary panels utilized for modelling cycle times. 96

List of Figures

Figure 1.1 Thesis organization beginning with creation of Computer-Aided Design (CAD) model 8
Figure 2.1. A sample floor framing model from BTL viewer 10
Figure 2.2. TJI® joist sizes available from Weyerhaeuser (Courtesy of: Weyerhaeuser) 14
Figure 2.3. Utilization factor for different joist materials [Feb 2016 to Sept 2016]........ 14
Figure 2.4. Components of stock cutting 15
Figure 2.5 Flowchart of dynamic waste reallocation (DWR) model 16
Figure 2.6 Construction of prefabrication structures in BTL format overview (I) Component overview, (II) File structure 17
Figure 2.7. Cutting processing (Courtesy of: design2machine.com) 18
Figure 2.8. Dynamic waste reallocation model matrix with corresponding parameters .. 22
Figure 2.9. A sample BTL [PART] containing a count of 15 lateral bracings, each with a length of 638.2 mm 26
Figure 2.10. (a) Waste allocation summary as produced by the cutting of stock material for a single floor design; (b) the overview of waste allocation cases numbered from Case 1 to Case 7 28
Figure 2.11. (a) Waste allocation summary as produced by the cutting of stock material for two separate floor designs; (b) the overview of waste allocation cases numbered from Case 1 to Case 7 28
Figure 2.12. (a) Waste allocation summary as produced by the cutting of stock material for four separate floor designs; (b) the overview of waste allocation cases numbered from Case 1 to Case 7 30
Figure 3.1 Overview of steel framing machine prototype 36
Figure 3.2 System Framework: Mapping for manufacturing residential walls (MMRW) 43
Figure 3.3 Computer-aided model of steel frame with corresponding parameters........ 44
Figure 3.4 Steel Framing Machine sketch, including machine parameters and direction of motion 45
Figure 3.5 Data parsing subroutine (A); BTL file interpreter process; and CSV file interpreter process (C) 46
Figure 3.6 Guided Nearest Neighbour (GNN) algorithm for screw-fastening 53

Figure 3.7 Screw-fastening logic for steel framing machine prototype (SFMP)............. 57
Figure 3.8 Steel framing machine working and buffer zones.. 62
Figure 3.9 Collision avoidance module 64
Figure 3.10 File structure of recipe file 65
Figure 3.11 Graphical user interface (GUI) for the proposed framework: main window (right) and motion control datasheet (left).. 66
Figure 3.12 Case study 1 summary table (A); top-down tool-path output; and recipe file output excerpt (C) 68
Figure 3.13 Case study 2 summary table (A); top-down tool-path output; and recipe file output excerpt (C) 69
Figure 3.14 Case study 3 summary table (A); top-down tool-path output; and recipe file output excerpt (C) 70
Figure 3.15 Case study 4 summary table (A); top-down tool-path output; recipe file output excerpt (C); and real-world operation (D) 72
Figure 4.1 Wood Framing Machine Prototype Overview 77
Figure 4.2 Sample multi-wood panel configuration.. 79
Figure 4.3 Wood Framing station with respective offsets 80
Figure 4.4 Data sorting for wood framing operations.. 81
Figure 4.5 Sample Format of “sortedDataList” 83
Figure 4.6 Wood Framing Machine’s file structure of recipe file 86
Figure 4.7 Sample GUI for wood framing machines... 87
Figure 5.1 Overview of Steel Framing Machine. 92
Figure 5.2 Main Logic for Discrete-event Simulation Model. 94
Figure 5.3 Simulation model of steel framing machine in Simphony.NET 94
Figure 5.4 Screw-fastening time comparison between simulation model and real-world prototype. 98
Figure 5.5 Comparison between time for production of 20 continuous panels and proposed scenarios. 98
Figure 5.6 Carriage movements versus time comparison to produce 20 wall panels using manual screw loading (i.e., scenario 1)... 99
Figure 5.7 Understanding effects on cycle time for varying feed rates on scenario 2...... 99

Figure 6.1 Research contributions and proposed future research 105

LIST OF ABBREVIATIONS

AEC	Architecture, Engineering, and Construction
BIM	Building Information Modelling
BTL	Building Transfer Language
CAD	Computer-aided Design
CAM	Computer-aided Manufacturing
CFS	Cold-Formed Steel
CSV	Comma-Separated Values
DES	Discrete-event Simulation
DWR	Dynamic Waste Reallocation
LGS	Light Gauge Steel
MMRW	Mapping for Manufacturing Residential Walls
GUI	Graphic User Interface
SFC	Screw-Fastening Carriage
SQL	Structured Query Language
PLC	Programmable Logic Controller

Chapter 1 Introduction

1.1 Background and Motivation

In 2017, Canada saw a boost in its residential construction sector with 219,763 housing starts (Statistics Canada 2018). And though residential construction is projected to level off in 2019, it is expected that housing starts will still remain close to 200,000 units/year (Housing Market Outlook 2017). Moreover, commercial construction is expected to grow by 6% in 2018, driven by a shift from a natural resources-based economy to a more high-tech, service-based economy (Oldcastle Business Intelligence 2017). To support the continuous demand for both residential and commercial construction, panelized construction is emerging as a cost-effective building solution (Liu 2016). Compared to conventional "stick frame" construction, panelized construction relocates the framing of building components to a factory environment, thus providing a controlled and efficient construction environment. Factory-built building components such as wall and floor panels allow for the implementation of a construction manufacturing philosophy, which offers a higher quality product with reduced site disruptions, faster construction, and improved safety. In fact, a study by Lawson and Ogden (2008) states that savings of 30-50% in total construction time can be achieved by applying modern off-site manufacturing methodologies. In the context of Canadian residential construction though, the full advantages of off-site construction are hampered by the limited availability of automated construction equipment and supporting processes.

To keep Canada's homebuilding industry competitive and to overcome the lack of automation, effective use of building information modelling (BIM) and innovative technologies must be enhanced to meet the full capability of panelized construction. Understanding the information generated by current BIM models and intelligently utilizing it can reduce material waste and provide systems to automate the light-frame manufacturing process. This research proposes an intelligent framework for computer-aided automation and optimization of light-frame building manufacturing. More specifically, this research promotes a novel approach for analyzing BIM models for floor structures to dynamically reduce wasted material by reallocating the potential cut waste to non-essential components, as well as utilizing BIM models to intelligently plan collision-free tool trajectories in the manufacture of light-gauge steel and timber panels. In addition to its use in testing collision-free tool trajectories on real-world machines, the data is deployed in discrete-event simulation (DES) for throughput time reduction analysis to further guide the development of automated machinery for light-gauge steel framing. The following sections provide detail information relating to each of the proposed research motivations.

1.1.1 Dynamic waste allocation

Construction industry generates a large quantity of waste due to ineffectual waste management systems and inefficient material utilization (Aldana & Serpell 2013). Though the growth of panelized construction has helped to reduce material waste by automating various processes that would require specialized carpentry knowledge if executed manually, there is still avoidable waste produced by machine cutting of the building materials. However, the existence of such waste arises from a reliance on accurate 3D BIM models that contain architecture, engineering, and construction (AEC) information. In other words, although the use of standard practices to create BIM models allows for fast turnover between jobs, such practices also result in unforeseen waste. This research focuses on developing an optimization model for maximizing the utilization of raw stock during the automatic cutting of floor components based on dynamic waste allocation (DWR). Although the cutting of floor components by means of optimized one-dimensional algorithms is a well-studied problem, the potential to reduce waste from floor manufacturing by understanding the shortcomings of standard practices is an avenue which remains mostly unexplored. Considering that panel prefabrication takes place in an indoor environment where waste material storage systems can be challenging to implement due to limited space and increased logistical challenges relating to materials and workflow, dynamic waste reallocation can provide positive results without requiring physical changes within the manufacturing facility. The proposed model reduces waste by reallocating otherwise wasted material to less critical components of the floor structure.

1.1.2 Tool-path generation method for automatic manufacturing of light-frame panels: Steel

The utilization of automated machines and Computer-Aided Design (CAD) / Computer-Aided Manufacturing (CAM) has allowed for rapid growth in the automotive and aerospace industries; however, the effects of such advancements have not been fully realized in construction to date. Numerically controlled machinery and integrated CAD/CAM software are essential tools in this regard (Lawson & Ogden 2008). In recent years there have been extensive efforts to automate construction operations by implementing varying degrees of automation. A study by Bock (2015) highlights a variety of single task construction robots (STCR) and their potential diffusion into built environments through future technological disruptions. Some literature relating to STCR does exist, such as steel beam assembly automation (Chu et al. 2013), contour crafting machine (Zhang & Khoshnevis 2013), and a bricklaying robot (Aguair & Behdinan 2015). However, no study available in the literature examines the automatic wall framing process.

To support the industrialization of off-site construction, a prototype machine is currently being tested at the University of Alberta that will assist in assembly and fastening of light-gauge steel framed wall-panels. In current practice, light-gauge steel walls are typically framed using manual operations, a practice which leads to lower productivity and to worker safety issues. Furthermore, even in the case of automated solutions, there are few studies available which address the challenge of transferring shop drawing information to machine-executable commands. This research proposes a framework for transferring such manufacturing information outputted from BIM models to an open-source file format readable by the programmable logic controller (PLC) used to control a steel framing machine prototype. The proposed framework allows for the generation of collision-free tool trajectories through the implementation of path generation and collision detection and avoidance modules. Such modules are designed to intelligently plan operations for safe tool movements and easy integration of in-shop modifications.

1.1.3 Tool-paths generation for automatic manufacturing of light-frame panels: Timber

As with light-gauge steel framing, a prototype machine is currently being tested at the University of Alberta that will support the industrialization of off-site timber construction by assisting in assembling and fastening of wood framed wall-panels. Chapter four expands on the research presented by Li (2016), providing a framework for transferring the manufacturing information outputted from BIM models to an open-source file format readable by the PLC used to control a wood framing machine prototype. The proposed framework allows for the generation of multi-spindle tool trajectories through the implementation of a methodology similar to the one proposed for light-gauge frame panel manufacturing.

1.1.4 Improved automated wall framing

Advances in off-site construction are promoting greater use of simulation modelling. DES is a computer-based simulation approach wherein real-world systems are converted to discrete events mimicking real-world processes. However, the development of simulation models within the domain of off-site construction industry has been mostly limited to scheduling and process improvement, with many studies applying DES to improve the production line for modular construction manufacturing (Altaf et al. 2014, 2015; Liu et al. 2015). To improve the overall production and efficiency of construction manufacturing, an in-depth understanding of sub-processes is critical. By individually optimizing each station within the production line, the overall processes can be furthered improved. This section describes the development of simulation models for the automated light gauge steel framing process using DES mimicking real-time machine production capacity and cycle time. At present, the literature on the development of such models for automated construction machinery is lacking; in this context, this section aims to showcase the advantages of simulation as a decision-making support tool. These models are a useful tool for identifying bottlenecks in machine operations that can be eliminated in order to improve productivity and meet local demand.

1.2 Research objectives

This research is built on the following hypothesis:

"The development of an integrated interpreter and optimizer system based on BIM model will support the material waste reduction, ease of data transfer, and safe tool-path generation for a robust light-framed panel manufacturing process, thereby allowing for faster and efficient production."

This research explores the use of BIM output to enhance information exchange between existing prototype machines with an aim to reduce waste and increase production capacity, respectively. This study focuses on developing methods for dynamically reallocating component cutting waste, tool-path generation for automatic manufacturing of light-frame panels, and improved automated wall framing process through simulation modelling. To validate this hypothesis, the following four research objectives are pursued:

1. Development of a model for maximizing the utilization of raw stock during the automatic cutting of floor joists, resulting in reduced material waste without requiring physical changes within the manufacturing facility.
2. Development of a framework for transferring manufacturing information, for both light-gauge steel and timber construction, from BIM to an open-source file format readable by the PLC used to control the prototype framing machine.

3. Development of collision detection and avoidance modules for generating collision-free tool trajectories for light gauge steel framing, while implementing multi-spindle path planning for wood framing machine.
4. Development of throughput time reduction techniques for gaining a more thorough understanding of manufacturing capacity and manufacturing time for automated light-framed construction machinery.

1.3 Organization of thesis

This thesis consists of six chapters. Chapter 1 presents a brief introduction to panelized construction and elaborates on existing limitations, research motivations, and objectives. Chapter 2 provides a literature review on the cutting-stock problem (CSP) and its relation to panelized construction. Then, an automated approach to dynamically reallocating otherwise wasted material to less critical components of the floor structure is presented. Chapter 3 describes a framework for construction CAM software for the automation execution of the steel framing process. Chapter 4 expands the framework as proposed in Chapter 3 to include automation execution of the wood framing process. Chapter 5 explains the limitations of current simulation modelling methods and proposes a DES model by which to further improve the light-gauge steel framing process. The relationships among these chapters is illustrated in Figure 1.1. Finally, Chapter 6 provides conclusions and summarizes the research contributions, limitations, and direction of future work.

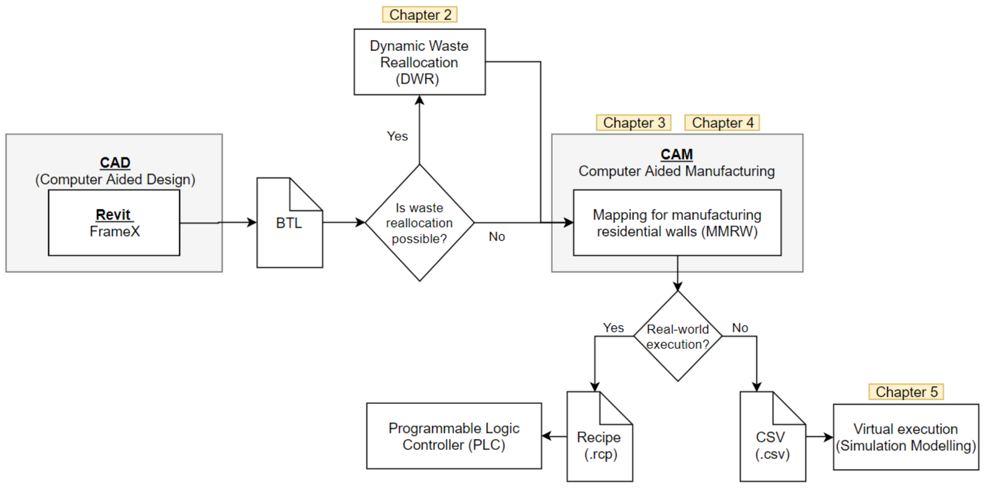

Chapter 2
Dynamic Waste Reallocation (DWR)
Yes
CAD
(Computer Aided Design)
Revit
FrameX
BTL
Is waste reallocation possible?
No
Chapter 3
Chapter 4
CAM
Computer Aided Manufacturing
Mapping for manufacturing residential walls (MMRW)
Yes
Real-world execution?
No
Programmable Logic Controller (PLC)
Recipe (.rcp)
CSV (.csv)
Chapter 5
Virtual execution (Simulation Modelling)

Figure 1.1 Thesis organization beginning with creation of Computer-Aided Design (CAD) model

Chapter 2 Dynamic waste allocation[1]

2.1 Introduction

The construction industry generates a large quantity of waste each year due to improper waste management systems and inefficient material utilization (Yeheyis et al. 2013). Extensive procedures have been proposed to reduce waste during the construction and demolition phases of a construction project, ranging from creating an inventory of discarded materials and evaluating their potential use to providing incentives to workers to save and exchange materials between all project stakeholders (Aldana & Serpell 2013). Furthermore, the growth of panelized construction has shifted traditional offsite production to a controlled factory environment, resulting in reduced construction cycle times and less material waste. However, even in panelized construction, where most of the construction takes place in an indoor environment, new waste material storage systems can be difficult to implement due to limited space and increased logistical challenges relating to material and workflow.

Since productivity and efficiency are of high priority, automated panelized construction companies rely on the creation of accurate 3D building information modelling (BIM) models that include architecture, engineering, and construction (AEC) information. For efficient construction of BIM models, standard practices are used to decrease the turnover between each job. This research describes the potential of reducing waste from floor manufacturing by understanding the shortcomings of standard practices and available literature. Notably, most of the cutting in an automated panelized construction company

[1] The manuscript appearing as Chapter 2 of this thesis was under review by the *International Journal of Construction Education and Research* at the time of publication of this thesis.

takes place internally; this research therefore proposes a model for predicting waste from the floor joist cutting process (Figure 2.1) and dynamically allocating the waste to lateral bracing prior to the actual cutting process. A greedy approach proposed by Avdzhieva et al. (2014) for stock cutting is incorporated to predict the cutting waste, whereas a detailed process is explained for dynamically acquiring floor design data from Building Transfer Language (BTL) machine files and reinterpreting the analyzed data into modified BTL files for automatic machine execution. Three case studies based on real scenarios are investigated and presented in this research in order to demonstrate the effectiveness of the proposed model.

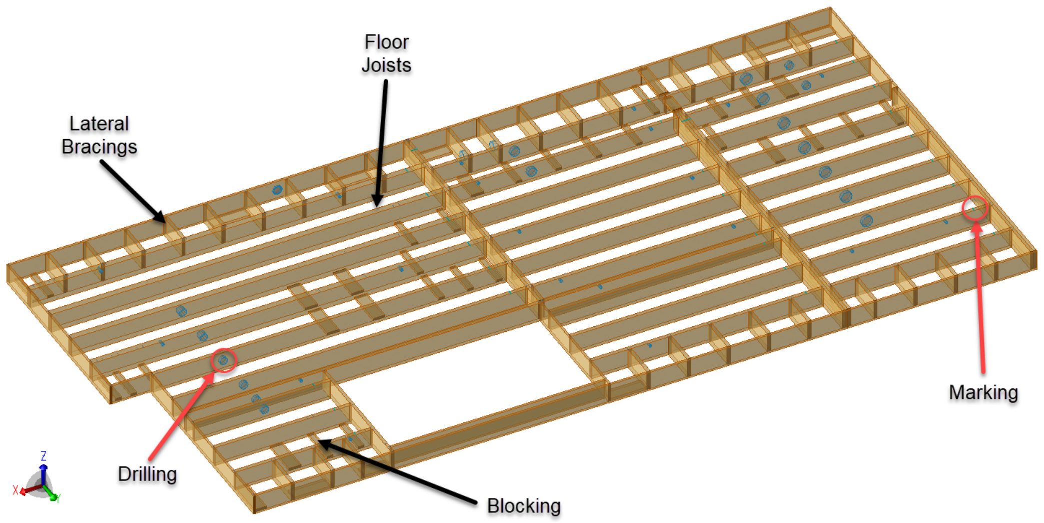


Figure 2.1. A sample floor framing model from BTL viewer

2.2 Literature review

The Cutting Stock Problem (CSP) is also known as the trim-loss problem (Cheng et al. 1994). In the CSP, raw materials are designed to be cut to a set of required sizes. It aims to minimize the material waste during the cutting process by developing the ideal cutting plan. It is a practical problem that has been extensively studied and sufficiently applied in the industry (Ragsdale & Zobel 2004). CSPs can have a range of dimensionalities depending on material types and applications within a given industry. The one-dimensional (1D) cutting problem usually involves raw materials such as steel or wood bars and wires, where only the length of stock is considered. Two-dimensional (2D) cutting occurs with sheathing materials and various shapes that need to fit together precisely in order to reduce the amount of leftover materials. Furthermore, spatial objects requiring three-dimensional (3D) cutting can be treated similarly to the packing problems (Wäscher et al. 2007).

For any given dimensionalities, the objectives of the CSP are universal, such as minimizing the leftover materials, minimizing the production cost, minimizing total number of cuts, etc. (Cherri et al. 2013). To solve the classic 1D CSP, Gilmore & Gomory (1961) transfer it to a linear programming (LP) problem. Heuristic methods like genetic algorithms and evolutionary algorithms have also been widely researched and applied to CSP (Hinterding & Khan 1995; Liang et al. 1998). Some algorithms for CSP, such as the next fit (NF) algorithm, the first fit (FF) algorithm, the best fit (BF) algorithm, and a combination of the three algorithms, are summarized and compared in the research carried out by Fayzrakhmanov et al. (2014). Most algorithms for reducing cutting patterns and minimizing trim losses are heuristic methods, such as the successive elimination method (Dikili et al. 2007), flexibility maximization problem (Wang & Liu 2014), and sequence-dependent cut losses (Garraffa et al. 2016).

Currently, panelized construction techniques are becoming well known, and the trend of applying this technique is rising. In panelized construction, prefabricated panels are produced in the factory and transported to be built on site (Weiss 2005). The panelization of the building process allows for the construction of modular walls, roof, and floors within a controlled manufacturing facility. CSP optimization can be adopted to minimize material waste during the prefabrication process thus making prefabrication construction an effective method of waste minimization (Tam et al. 2005; Tam et al. 2006). Liu (2016) developed a BIM-based approach to design the layouts of drywall and sheathing in the panelized production process with minimized material waste by incorporating CSP optimization. Zhao (2015) presented a methodology to optimize the usage of lumber and sheathing in panelized construction manufacturing. The optimized cutting plan utilizes simplex and greedy algorithms for the 1D CSP of lumber, as well as greedy and bottom-left heuristic algorithms for the 2D CSP of sheathing.

Some constraints are applied to the CSP because they are related to the production cost, for example, the amount of open stock representing utilized stock pieces. Therefore, several studies have focused on solving the ordered CSP by considering the specific number of pieces of open stock. To control the number of partially-completed jobs, Ragsdale & Zobel (2004) presented a genetic algorithm solution to accommodate the ordered CSP while continuing to reduce the leftover materials. The greedy randomized adaptive search procedure (GRASP) metaheuristic is also developed to deal with the ordered CSP. This research has pioneered the development of a method to obtain an accurate result in mass-ordered cases (Rabello Golfeto et al. 2008).

Larger cutting scraps can be reutilized as raw material. Sinuany-Stern & Weiner's (1994) research on CSP includes two objectives: (1) to minimize the trim loss of the cutting, and (2) to maximize the leftovers from the cutting of previous stock such that it can be reused. Cherri et al. (2013) proposed a modified heuristic approach for the CSP with usable leftovers, assuming that the use of usable leftovers takes priority over the raw material. The results prove this approach effectively produces less waste than the usual CSP procedure. Cui & Yang (2010) proposed a combined algorithm, the residual sequential heuristic procedure (RSHP), to solve the CSP with usable leftovers.

2.3 Waste minimization: Floor Manufacturing

For normal operations floor structures are constructed from materials with varying strengths, for instance, lateral bracing primarily uses lower capacity joists, i.e., TJI-210, and floor joists use stronger materials, i.e., TJI-360 or TJI-560 (Figure 2.2). Since lateral bracing provides structural stability and helps resist lateral loads, each floor structure must comply with standard practices; however, there exists an opportunity to replace lateral bracing with stronger substitutes, i.e., TJI-360 & 560 joists. The contribution of this research is to demonstrate that by reallocating waste from stronger substitute materials to lateral bracing, which does not affect the structural stability of the floor structure, the overall material utilization can be improved.

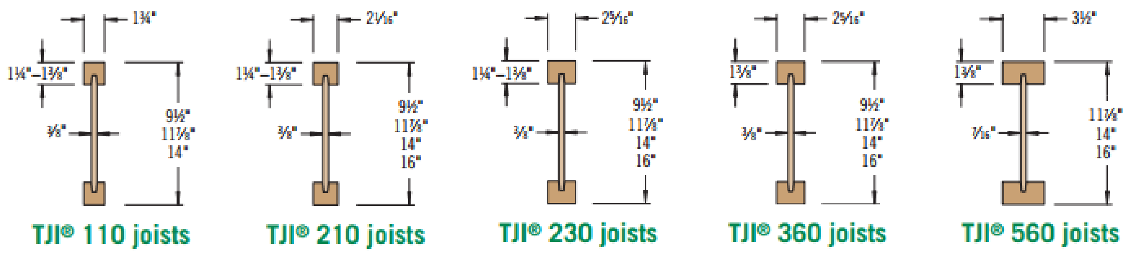


Figure 2.2. TJI® joist sizes available from Weyerhaeuser (Courtesy of: Weyerhaeuser)

To properly understand the potential of dynamically utilizing waste, multiple site visits are conducted, and numerous data sets are obtained from ACQBUILT, Inc., a prefabricated panelized construction company based in Edmonton, Canada. Through the initial investigation, it is found that TJI-360 & 560 have on average a lower utilization factor as compared to TJI-210 (Figure 2.3). Given that most lateral bracing is cut from TJI-210, there is a greater possibility of increased stock utilization by utilizing waste from TJI-360 & 560, which in turn will reduce the open stock requirements for lateral bracing.

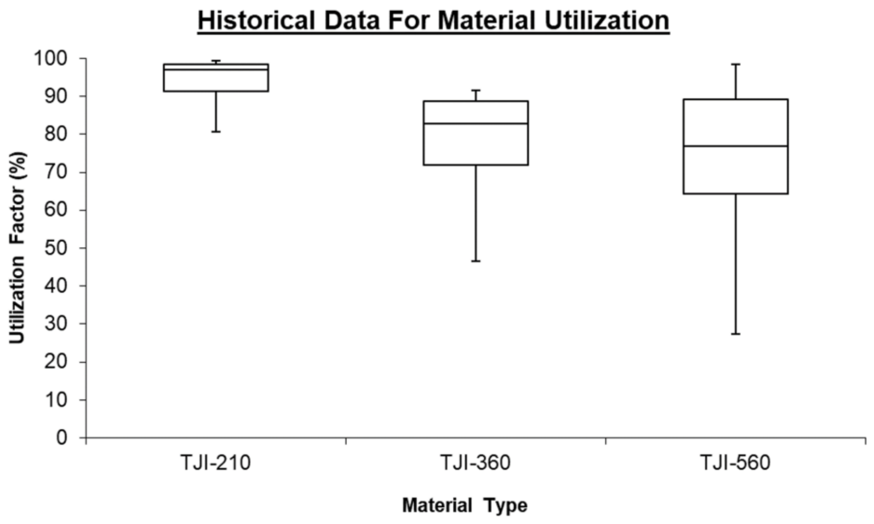


Figure 2.3. Utilization factor for different joist materials [Feb 2016 to Sept 2016]

The cutting of various stock lengths results in two primary types of waste, unavoidable and avoidable, arising from the cutting process and inefficient cutting patterns, respectively. Even with the optimum cutting pattern, minimum waste generation is dependent upon the saw thickness and leftover stock. During automated cutting, extra waste is also generated by the support jaw (grip) as presented in Figure 2.4. Due to the lack of customization of proprietary software for the automated cutting machine, the research focuses on the optimization of leftover stock (W_i), before actual cutting, rather than optimizing the cutting patterns.

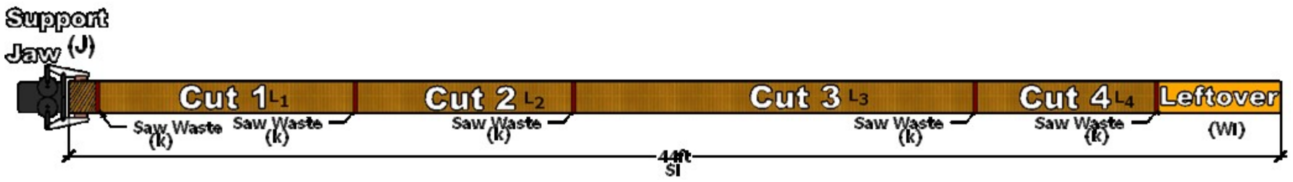


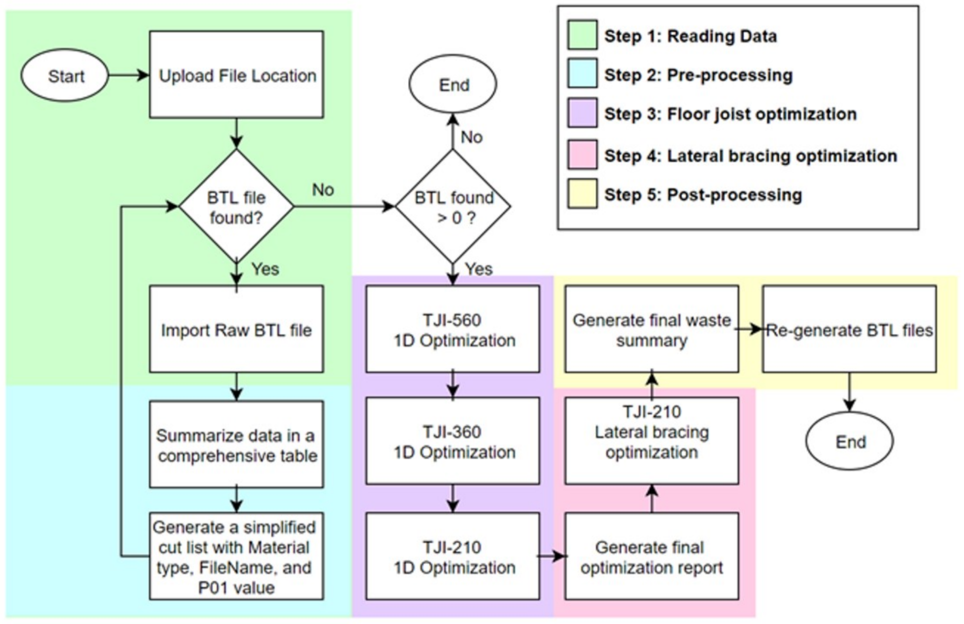
Figure 2.4. Components of stock cutting

The contributions of this research encompass the following areas: (1) increase of material utilization without any capital expenditure; (2) the ease of usability and flexibility for immediate implementation in a factory environment; (3) the development of a centralized system with an ability to read, optimize, and reconstruct multiple BTL files.

The gap between current industry practice and available automated cutting technology results in limited flexibility to accomplish the above objectives. Machines are designed to follow built-in algorithms which optimize the cutting of each piece of stock depending on its material type, whereas industry practice does not take into account the potential of dynamic waste reallocation. The successful implementation of the aforementioned research areas within a single encompassing model will contribute to a reduction in material waste and will showcase the potential of proactive waste reduction in panelized construction.

2.4 Model: Dynamic Waste Reallocation

The components of the dynamic waste reallocation (DWR) model for panelized floor cutting are presented in Figure 2.5. These components will be discussed in the following sub-sections.

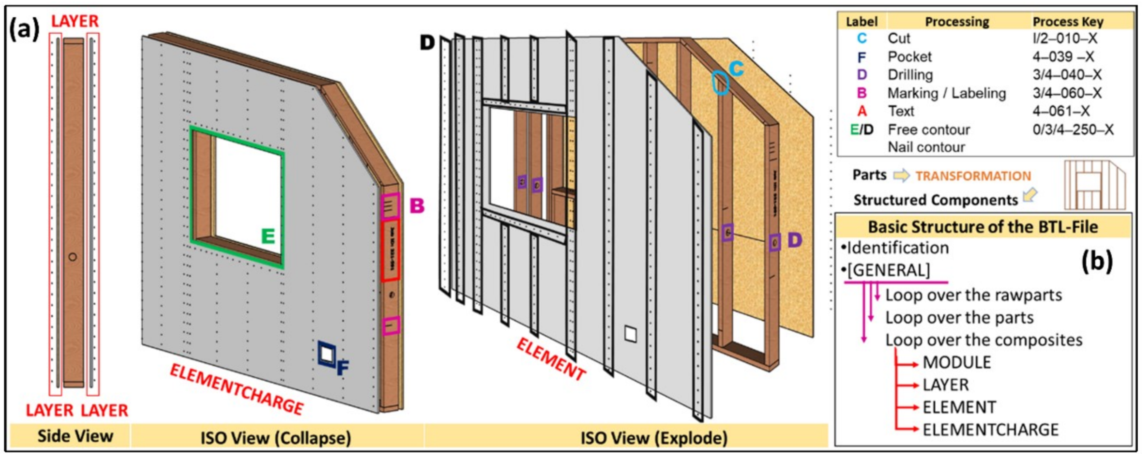


Step 1: Reading Data
Step 2: Pre-processing
Step 3: Floor joist optimization
Step 4: Lateral bracing optimization
Step 5: Post-processing
Start
Upload File Location
BTL file found?
No
Yes
Import Raw BTL file
Summarize data in a comprehensive table
Generate a simplified cut list with Material type, FileName, and P01 value
End
BTL found > 0 ?
No
Yes
TJI-560 1D Optimization
TJI-360 1D Optimization
TJI-210 1D Optimization
Generate final optimization report
TJI-210 Lateral bracing optimization
Generate final waste summary
Re-generate BTL files
End

Figure 2.5 Flowchart of dynamic waste reallocation (DWR) model

2.4.1 Initialization

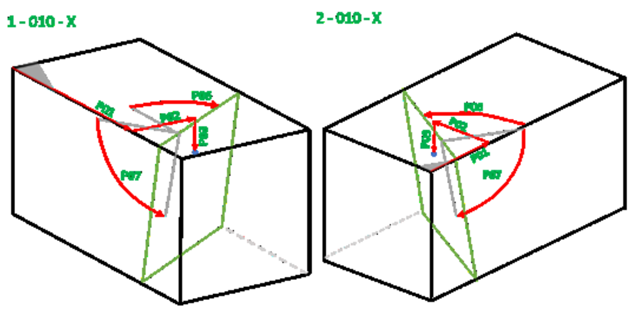
The first step for implementing the proposed model is to understand the BTL file format since it describes parts in a machine-independent geometry format and, as such, a complete understanding of the format and all “processings” is required. The BTL file offers a standard interface for communicating manufacturing data from design software to cutting machine (see Figure 2.6b). Each BTL file contains individual parts representing basic building blocks (e.g., studs, blocking, bracings) and through transformation, each part is oriented as per the real-world design. Moreover, additional prefabrication information (e.g., nailing, labeling, routing) is also included in the BTL file, which allows for easy communication between BIM models and manufacturing machines (Figure 2.6a). An in-depth guide for BTL format is available at design2machine.com. As presented in Figure 2.7, for the cutting process, parameters P01, P02, P03, P06, and P07 are used to define standard cutting information. Furthermore, the complete DWR model is programmed using Visual Basic for Applications (VBA) within Microsoft® Excel. Here, a standalone Excel-based solution can be easily implemented without any capital expenditure as required per the project's objective.



Label	Processing	Process Key
C	Cut	I/2–010–X
F	Pocket	4–039 –X
D	Drilling	3/4–040–X
B	Marking / Labeling	3/4–060–X
A	Text	4–061–X
E/D	Free contour Nail contour	0/3/4–250–X

Basic Structure of the BTL-File
- •Identification
- •[GENERAL]
  - Loop over the rawparts
  - Loop over the parts
  - Loop over the composites
    - MODULE
    - LAYER
    - ELEMENT
    - ELEMENTCHARGE

Figure 2.6 Construction of prefabrication structures in BTL format overview (I) Component overview, (II) File structure



where

P01-	Distance from start to the reference point
P02-	Distance from reference edge to the reference point
P03-	Distance from reference side to the reference point
P06-	Angle between reference edge and cut edge
P07-	Inclination between face and reference side

Figure 2.7. Cutting processing (Courtesy of: design2machine.com)

2.4.2 Reading Data

The data reading process begins with a pre-defined folder path containing all the required BTL files. Due to rapid turnaround times in panelized construction factories, multiple floors can be constructed per day, thus requiring multiple BTL files. The files are read using a “while” loop, which loops through all the files contained within the specified folder until the condition, “no more files”, is met. Furthermore, after locating the required BTL file, the data from the given file is imported and is stored for pre-processing as shown:

$$DS_k = \{\forall_i = \{1, \dots, n), l_i\} \tag{2.4.2.1}$$

where

- DS_k = Data storage array with “k” representing file index
- l_i= i$^{\text{th}}$ line in a given BTL file with “n” representing last line in specified BTL file

2.4.3 Pre-Processing

The pre-processing of each file takes place after each file (DF_k) is imported. The exact material for each lateral bracing piece (length = 610 mm) and floor joist is summarized in four nested arrays (see Equation 2.4.3.1), including the necessary length (L_i), required quantity (Q_i), file name (FN_i), and part number (PT_i). The part number (known as [PART] in BTL file format) is a unique ID assigned to each unique item within a given file. Here, the FN_i and PT_i uniquely define the properties of each cut list item, which is crucial for the reconstruction of each file as outlined in the post-processing section of this research. It should be noted that most lateral bracing and certain floor joists are of similar lengths, hence requiring Q_i having a value greater than one.

$$\begin{cases} LB_k = \{\forall i = \{1, \dots, size(DS_k)\}, (PT_i, MT_i, FN_i, L_i, Q_i) \in DS_k \mid MT_i = \text{"TJI-210"} \wedge L_i < 610\} \\ FJ_k = \{\forall i = \{1, \dots, size(DS_k)\}, (PT_i, MT_i, FN_i, L_i, Q_i) \in DS_k \mid MT_i = \text{"TJI-210"} \wedge L_i \geq 610\} \\ FJ3_k = \{\forall i = (1, \dots, size(DS_k)), (PT_i, MT_i, FN_i, L_i, Q_i) \in DS_k \mid MT_i = \text{"TJI-360"}\} \\ FJ5_k = \{\forall i = \{1, \dots, size(DS_k)\}, (PT_i, MT_i, FN_i, L_i, Q_i) \in DS_k \mid MT_i = \text{"TJI-560"}\} \end{cases} \tag{2.4.3.1}$$

where

- LB_k (mm) = List of all lateral bracings in kth file
- FJ_k, $FJ3_k$, $FJ5_k$ (mm) = List of floor joists in kth file and having a material type of TJI-210,360,560, respectively

2.4.4 Floor joist Cutting

After quantifying and analyzing the actual material data from the cutting machine, the real-world cutting process is coordinated with a simplified greedy algorithm that provides a one-to-one relationship between simulated and actual data. The heuristic cutting approach follows a greedy algorithm as explored by Avdzhieva et al. (2014) and involves two key steps: (1) sort the cutting lengths (L_i) in descending order; and (2) cut the largest lengths among the remaining stock pieces until all the items are cut. Due to its simple nature, this approach considerably reduces the complexity of the CSP and rapidly provides an approximate solution by locally optimizing the cutting patterns. More specifically, the overall goal is to ensure the quantity required (Q_i) is same as quantity produced (c_i) for each cut list item i as shown in Equation 2.4.4.1, where c_i is the summation of the quantity of item i cut from available length j (Equation 2.4.4.2). Since the primary goal is to achieve Q_i quantities, new raw lengths must be acquired if $c_i \neq Q_i$ as illustrated by Equation 2.4.4.3. Finally, available useable length (AL_j) of stock item j is calculated by subtracting stock length (S_j) from jaw wastage (J) as shown in Equation 2.4.4.5, thus requiring total cut items $q_{i,j}$ with length (L_i) to have a cumulative length less than AL_j (Equation 2.4.4.4). The final leftover waste (W_j) is calculated as illustrated in Equation 2.4.4.6.

$$Q_i = c_i, \forall_i = \{1, \dots, n\} \tag{2.4.4.1}$$

$$c_i = \sum_{j=1}^{m} q_{i,j}, \forall_i = \{1, \dots, n\} \tag{2.4.4.2}$$

$$\begin{cases} (k + L_i) \times (Q_i - c_i) \text{ mm length required if } c_i \neq Q_i \\ 0 \text{ mm length required otherwise} \end{cases}, \forall_i = \{1, \dots, n\} \tag{2.4.4.3}$$

$$\sum_{i=1}^{n} q_{i,j} \times L_i + k \times q_{i,j} \leq AL_j, \forall_j = \{1, \dots, m\} \tag{2.4.4.4}$$

$$AL_j = S_j - J, \forall_j = \{1, \dots, m\} \tag{2.4.4.5}$$

$$W_j = AL_j - \left(\sum_{i=1}^{n} q_{i,j} \times L_i + k \times q_{i,j}\right), \forall_j = \{1, \dots, m\} \tag{2.4.4.6}$$

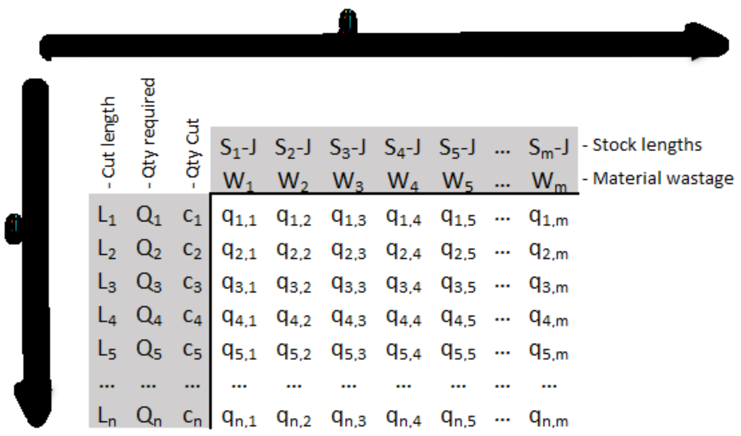
where

- AL (mm)= Available length from item j
    - m = total number of raw pieces, either new or leftover stock lengths, j = 1,…, m
- L_i (mm) = Length require for cutting item i, i = 1, … , n
    - n = total number of unique cut lengths
- Q_i = Quantity require for item i
- S_j (mm) = Stock length of item j, j = 1, … , m
- $q_{i,j}$ = Quantity of item i cut from available length j
    - $q_{i,j} \geq 0$ integer, i = 1, … , n; j = 1, … , m
- W_j (mm) = Leftover length from item j
    - $W_j \geq 0, j = 1, \dots, m$
- J (mm) = Jaw wastage = 20 (constant)
- k (mm) = Saw thickness = 5.2 (constant)
- c_i = Quantity produce for item i

To construct the floor joist cutting summary, waste generation from each material type is investigated separately by utilizing a matrix structure as presented in Figure 2.8. The pseudo-code for translating above model to VBA structure is presented below, wherein the quantity of item i cut from stock j ($q_{i,j}$) is generated by the greedy approach as follows:

```
- Sort cutting lengths in descending order, such that $L_1 > L_2 > \dots > L_n$
```

- *Start from j = 1 and set the stock length to S_1 and calculate available length by subtracting saw wastage (J) from the original stock length. Set waste (W_1) to modified stock length.*
    - *Start from i = 1 and increment i until an item i is found such that $L_i + k \leq W_1$*
    - *Check:*
        - *If $Q_i \geq 1$, set $q_{i,1}$ to the maximum $q_{i,1}$ such that $q_{i,1}(L_i + k) \leq W_1$ and $\sum_{j=1}^{m} q_{i,j} \leq Q_1$*
        - *If $Q_1 = 1$, set $q_{i,1} = 1$*
    - *Update the waste (remainder stock length) $W_1 = W_1 - q_{i,1}(L_i + k)$*
    - *Update $c_i = \sum_{j=1}^{m} q_{i,j}$ – denoting the total of stock material pieces split between various pattern j*
    - *Repeat above process until i equals n*
- *If stock cutting remains after looping through all required lengths, e.g., $\sum_1^n Q_i \neq \sum_1^n c_i$, increment a new stock such that j=j+1 and repeat the process of finding all the remaining $q_{i,j}$ values.*



- Cut length	- Qty required	- Qty Cut	S_1-J W_1	S_2-J W_2	S_3-J W_3	S_4-J W_4	S_5-J W_5	... ...	S_m-J W_m	- Stock lengths - Material wastage
L_1	Q_1	c_1	$q_{1,1}$	$q_{1,2}$	$q_{1,3}$	$q_{1,4}$	$q_{1,5}$	...	$q_{1,m}$	
L_2	Q_2	c_2	$q_{2,1}$	$q_{2,2}$	$q_{2,3}$	$q_{2,4}$	$q_{2,5}$	...	$q_{2,m}$	
L_3	Q_3	c_3	$q_{3,1}$	$q_{3,2}$	$q_{3,3}$	$q_{3,4}$	$q_{3,5}$	...	$q_{3,m}$	
L_4	Q_4	c_4	$q_{4,1}$	$q_{4,2}$	$q_{4,3}$	$q_{4,4}$	$q_{4,5}$	...	$q_{4,m}$	
L_5	Q_5	c_5	$q_{5,1}$	$q_{5,2}$	$q_{5,3}$	$q_{5,4}$	$q_{5,5}$	...	$q_{5,m}$	
...	...	...	...	...	...	...	...	...	...	
L_n	Q_n	c_n	$q_{n,1}$	$q_{n,2}$	$q_{n,3}$	$q_{n,4}$	$q_{n,5}$	...	$q_{n,m}$	

Figure 2.8. Dynamic waste reallocation model matrix with corresponding parameters

It should be noted that the VBA function, as mentioned above, will loop through "n" cut list items and "m" open stocks by utilizing the "do-while" loop, which will add new stock until all the cutting has been achieved after which the loop will terminate.

By following the above algorithm, a visual matrix for each material type can be constructed with corresponding waste from each stock piece. As mentioned in Section 2.4.3, the file name and part numbers that uniquely define the properties of each cut list item are also transferred throughout the optimization process.

2.4.5 Lateral Bracing Cutting

For optimal material usage, the lateral bracing is cut beginning with waste material produced by TJI-210, TJI-360, and TJI-560, respectively. The resulting waste reallocation process means that no reallocation would be required if all the bracing pieces can be produced from the default TJI-210 material waste. Before implementing the lateral bracing optimization process, the summary of each piece of utilized stock is generated to showcase the overall utilization of stock (see Table 2.1).

Table 2.1 Sample summary table for optimizing floor joist cutting (AS = allocated stock, mm)

(1)	(2)	(3)	(4)	(5)	(6)	(7)
Stock Type	No. of stock	Post Opt. Scrap (mm)	Raw Stock (mm)	% Utilization	Post Lt. Opt. Scrap (mm)	% Final Utilization
TJI-210, TJI-360, TJI-560	1	W_1	S_1	$\left[1-\frac{(3)}{(4)}\right]\times 100$	W_1-AS_1	$\left[1-\frac{(6)}{(4)}\right]\times 100$
	2	W_2	S_2		W_2-AS_2	
	.	.	.		.	
	.	.	.		.	
	.	.	.		.	
	m	W_m	S_m		W_m-AS_m	

The algorithm for lateral bracing cutting remains identical to that of floor joist cutting; however, changes arise from varying stock lengths (Sj), representing the actual leftover material (W_j) from the floor joists cutting. Furthermore, considering the possible lack of available waste material, the model contains an extra script which adds the ability to include new standard TJI-210 stock as required. Finally, a summary of the optimized lateral bracing is presented in Table 2.1 under the headings, “Post Lt. Opt. Scrap” and “% Final Utilization”.

The completion of the lateral bracing optimization phase produces one of seven possible cases as presented in Table 2.2. In the first three cases, the original [PART] remains intact, whereas, for the last four cases, the original [PART] is split into one or more separate parts. For instance, if a given series of lateral bracing requires ten similar pieces from TJI-210, and after optimization it is concluded that five pieces can be obtained from TJI-210, with two from TJI-360, and the final three from TJI-560, this case will be marked with the case 7 code (Table 2.2). Proper assignment of these cases is critical since these unique codes are used to reconstruct the modified BTL file as presented in the following section.

Table 2.2. Summary of cases produced throughout the optimization process

Case	Material Split	Definition
1	210	No change required
2	360	Change [PART] to TJI-360 from TJI 210
3	560	Change [PART] to TJI-560 from TJI 210
4	210+360	Split the [PART] between TJI 210 and TJI 360 depending on $q_{I,j}$ values
5	210+560	Split the [PART] between TJI 210 and TJI 560 depending on $q_{I,j}$ values
6	360+560	Split the [PART] between TJI 360 and TJI 560 depending on $q_{I,j}$ values
7	210+360+560	Split the [PART] between TJI 210, TJI 360, and TJI 560 depending on $q_{I,j}$ values

2.4.6 Post-Processing

Excluding the first case, each other situation requires a change from the original BTL file depending on the corresponding material split. To maintain the original BTL file format, post-processing is programmed in VBA such that all the machining operations and other properties relating to each part remain consistent. For cases 2 and 3, the material type is switched from TJI-210 to TJI-360 and TJI-560, respectively. Here, the change in material type only requires changing material properties (i.e., width, height, etc.), material type, and designation, whereas, for cases 4 to 7, one or two new parts are created as per the material split. Splitting one part into multiple parts requires changing multiple fields, such as "SINGLEMEMBERNUMBER", "ASSEMBLYNUMBER", "MATERIAL", and "ENDOFFSET", among others as listed in Figure 2.9. It should be noted that the "COUNT" field will be split between different materials; however, the final cumulative count will remain same as per the original part. The properties for 3D models (such as "TRANSFORMATION" and "UID") are also split as per the material split. As a reference, the lateral bracing part structure in BTL file format is presented in Figure 2.9.

```
[PART]
SINGLEMEMBERNUMBER: 38
ASSEMBLYNUMBER:    "38"
ORDERNUMBER:       0
DESIGNATION:       "- / 20GLR-16-0010 / TJI 210 (11 7I8)"
ANNOTATION:        ""
STOREY:            "38"
GROUP:             "01"
STOREYTYPE:        CEILING
ELEMENTNUMBER:     "E102"
LAYER:             0
MODULENUMBER:      "BST = 00000;P1P2;"
PACKAGE:           "E102"
MATERIAL:          "TJI 210 (11 7I8)"
TIMBERGRADE:       ""
QUALITYGRADE:      ""
COUNT:             15
LENGTH:            00006382
HEIGHT:            00003016
WIDTH:             00000524
PLANINGLENGTH:     00000000
STARTOFFSET:       00000200
ENDOFFSET:         00000200
UID:               3885
TRANSFORMATION:    OX:00009668  OY:00000318  OZ:0000-222  XX:00000000  XY:00010000  XZ:00000000  YX:00000000
YY:00000000  YZ:00-10000
….
UID:               3901
TRANSFORMATION:    OX:00100584  OY:00000318  OZ:0000-222  XX:00000000  XY:00010000  XZ:00000000  YX:00000000
YY:00000000  YZ:00-10000
PROCESSINGQUALITY:  AUTOMATIC
PROCESSKEY:  2-010-4       Cut
PROCESSPARAMETERS:  P01:00000000  P02:00000000  P03:00000000  P06:00000900  P07:00000900
PROCESSIDENT: 1
PROCESSKEY:  1-010-4       Cut
PROCESSPARAMETERS:  P01:00006382  P02:00000000  P03:00000000  P06:00000900  P07:00000900
PROCESSIDENT: 2
```

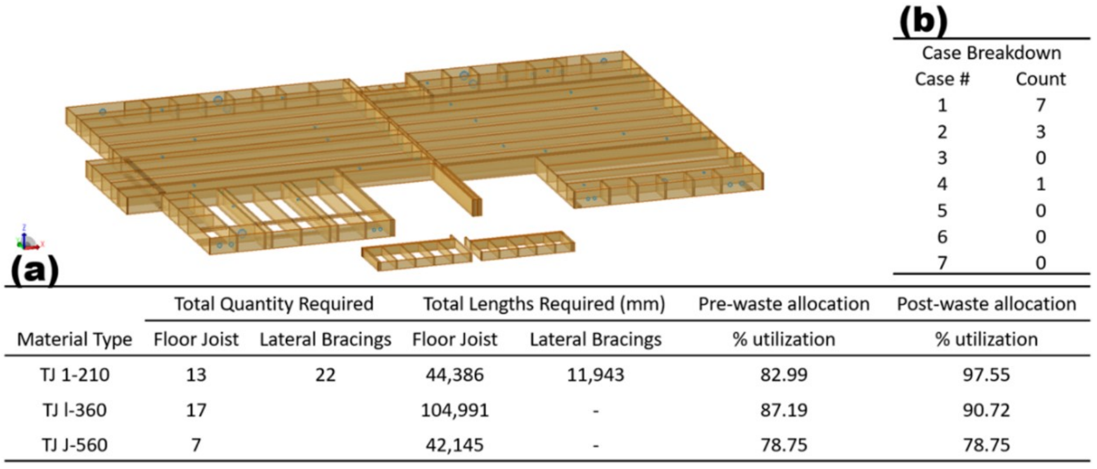
Figure 2.9. A sample BTL [PART] containing a count of 15 lateral bracings, each with a length of 638.2 mm

2.5 Case Studies

Several design cases, supplied by ACQBUILT, Inc., an Edmonton-based prefabricated home building company, are studied to validate the applicability of the DWR model. The aim of these studies is to compare the waste generated through the normal daily procedures as compared to waste produced by applying the new waste reduction model. The cutting of floor components is simulated using commercial simulation software designed specifically for the onsite cutting machine. These simulations include obtaining the waste generated from normal BTL as compared to the modified BTL file. The following case studies are analyzed using a Windows 10 computer with Intel Core i7-4790 CPU, and 16 GB RAM.

3.1.1 Case Study I

For the first case study, the production of a single floor plan is investigated. A brief description of the single floor panel is presented in Figure 2.10a, where a total of 22 lateral bracing pieces are required with a cumulative length of 11,943 mm (39.18 ft). During the analysis, it is noted that the material utilization of TJI-210 increases from 82.99% to 97.55%, meaning that all the waste produced from the TJI-210 floor joists can be re-allocated to the cutting of lateral bracing. However, since not all the lateral bracing can be cut from TJI-210 waste, the remaining lateral bracing could be obtained from TJI-360 waste, thereby increasing the utilization of TJI-360 from 87.19% to 90.72%, amounting to a cumulative length of 4,258 mm (13.97 ft). In other words, rather than obtaining new stock (i.e., the standard size of 44 ft), the TJI-360 waste can be utilized. This can be observed in Figure 2.10b, where cases 2 and 4 present the number of lateral bracing pieces requiring material type reassignment. This investigation can be optimized in 20.31 seconds and requires an additional 0.95 seconds to generate the modified BTL file.



Case Breakdown	
Case #	Count
1	7
2	3
3	0
4	1
5	0
6	0
7	0

	Total Quantity Required		Total Lengths Required (mm)		Pre-waste allocation	Post-waste allocation
Material Type	Floor Joist	Lateral Bracings	Floor Joist	Lateral Bracings	% utilization	% utilization
TJ 1-210	13	22	44,386	11,943	82.99	97.55
TJ I-360	17		104,991	-	87.19	90.72
TJ J-560	7		42,145	-	78.75	78.75

Figure 2.10. (a) Waste allocation summary as produced by the cutting of stock material for a single floor design; (b) the overview of waste allocation cases numbered from Case 1 to Case 7

3.1.2 Case Study II

For the second case study, the parallel production of two individual floor plans as presented in Figure 2.11a is investigated, where a total of 61 lateral bracing pieces of various lengths, amounting to a cumulative length of 145,910 mm (478.71 ft), are required. After running the optimization model, it is noted that the material utilization of TJI-210 increases from 83.87% to 99.37%, which, due to the need for additional lateral bracing, results in increased utilization of TJI-360 from 58.88% to 64.38%. The increased utilization of TJI-360 correlates to the reallocation of 4,821 mm (15.82 ft) of potential waste; hence, reducing the requirement of one additional TJI-210 standard stock piece spanning a standard length of 44 ft. The breakdown of waste allocation can be observed in Figure 2.11b, where case 4 highlights the number of lateral bracing pieces requiring material type reassignment. This investigation can be optimized in 19.83 seconds and requires an additional 1.48 seconds to generate the modified BTL files.

(b)

Case Breakdown

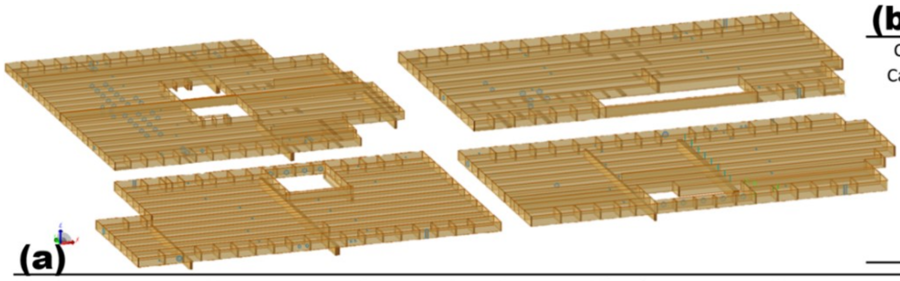
Case #	Count
1	11
2	0
3	0
4	2
5	0
6	0
7	0

	Total Quantity Required		Total Lengths Required (mm)		Pre-waste allocation	Post-waste allocation
Material Type	Floor Joist	Lateral Bracings	Floor Joist	Lateral Bracings	% utilization	% utilization
TJ 1-210	19	61	145,910	31,488	83.87	99.37
TJ I-360	8		64,110	-	58.88	64.38
TJ J-560	1		7,772	-	58.08	58.08

Figure 2.11. (a) Waste allocation summary as produced by the cutting of stock material for two separate floor designs; (b) the overview of waste allocation cases numbered from Case 1 to Case 7

3.1.3 Case Study III

For compressed project schedules, multiple houses can be constructed within a matter of days; hence, this case study is conducted to understand the potential savings from larger jobs. Simultaneous production of four individual floor plans (presented in Figure 2.12a) requires the production of a considerable number of joists and bracing pieces, which in turn improves the overall utilization of various materials; however, as in previous cases, the allocation of leftover waste can further improve the utilization of each material type. It should be noted that after fully utilizing all of the waste of TJI-210, the TJI-360 waste is also fully used, resulting in reducing the requirement of TJI-210 by 5,007.6 mm (16.43 ft). Furthermore, due to the significant amount of lateral bracing required, the waste produced from TJI-560 could also provide multiple lateral bracing pieces with a cumulative length of 42,580 mm (139.70 ft). The total waste allocation from TJI-210 to TJI-360 and TJI-560 amounts to a cumulative length of 47,588 mm (156.13 ft), which results in a savings of four TJI-210 standard stock pieces. Similar to the previous cases, the breakdown of each case can be observed in Figure 2.12b, where most of the cases have a non-zero count due to the complex nature of this case study. Finally, it should be noted that since most of the waste material is fully utilized, any increase in the amount of necessary lateral bracing would require the addition of new TJI-210 stock. This investigation requires 24.84 seconds to optimize and a further 3.07 seconds to generate the modified BTL files.



(b)

Case Breakdown	
Case #	Count
1	13
2	0
3	9
4	0
5	1
6	0
7	2

	Total Quantity Required		Total Lengths Required (mm)		Pre-waste allocation	Post-waste allocation
Material Type	Floor Joist	Lateral Bracings	Floor Joist	Lateral Bracings	% utilization	% utilization
TJ 1-210	46	122	280,368	72,537	91.11	99.41
TJ I-360	4	0	48,514	-	90.61	99.96
TJ J-560	23	2	217,964	914	81.77	97.67

Figure 2.12. (a) Waste allocation summary as produced by the cutting of stock material for four separate floor designs; (b) the overview of waste allocation cases numbered from Case 1 to Case 7

3.1.4 Summary of case studies

From the three case studies outlined above, it can be observed that at least one standard TJI-210 stock piece can be saved by implementing the DWR model, which can amount to lower material cost and reduced waste. In term of actual cost savings, reduction in one TJI-210 stock piece provided $88 in material savings, which combined with the economy of scale of the local manufacture is estimated to provide an annual savings of $79,200. This is especially useful since this process can be run within a matter of seconds and only requires Excel software to execute. Furthermore, as additional floor plans are executed in parallel, the potential savings can lead to savings of multiple TJI-210 stock pieces. A limitation for sorting pieces arises when numerous jobs are executed in parallel; however, in the future, such a limitation can be overcome using computer-vision technology (Ahmad et al. 2016, 2017; Ahmad & Plapper 2015). Nevertheless, the local manufacturer involved in this project confirms the savings as expressed in the three case studies and states the ease of implementing and utilizing the above process.

The implementation of the DWR model is shown to require less than 30 seconds to dynamically read, optimize, and reconstruct up to four separate BTL files and provides a positive impact on the material utilization, thereby achieving the research objectives by providing a dynamic system for reducing waste without requiring implementation of complex design changes within existing workflow. The successful testing of the DWR model showcases the advantages of proactive waste reduction strategies, which have been lacking in the domain of panelized construction.

2.6 Conclusion

Growth in panelized construction has promoted the development of innovative processes, resulting in reduced waste and lower construction costs. However, among the manufacturing processes in panelized construction, there is reduceable waste produced by the machine cutting of building materials. Due to limited literature on the SCP within panelized construction, a new model for maximizing the utilization of raw stock during the automatic cutting of floor components is presented in this research. The proposed model, using the heuristic cutting approach, can reduce material waste by reallocating otherwise wasted material to less critical components of the floor structure. However, due to proprietary nature of the commercial equipment, any modification to existing equipment is highly complex if not impossible, therefore using historical data from a panelized construction homebuilder, the proposed model is designed to accurately predict the cutting outcome before the real-world commencement of the cutting phase. Understanding the aforementioned cutting outcome thus allows for the reallocation of potential waste, which is the basis of this research.

The presented case studies demonstrate the advantage of using such a system in an actual factory setting. In fact, it is shown that implementing the proposed model reduces demand for at least one piece of TJI-210 stock (i.e., 44-ft standard stock piece); however, the stock demand can be further reduced when executing multiple parallel cutting jobs. The cost saving of one stock piece was estimated to $88, which due to the economy of scale is expected to save the case company $79,200 annually. Furthermore, due to its ease of usability, such a model can easily be implemented in a factory setting without requiring any capital expenditure and without negatively impacting the overall workflow. Future work will focus on storing cut pieces when executing multiple parallel jobs, and integration of integrating the proposed system into other related cutting applications where the design allows for the reallocation of leftover waste.

Chapter 3 Tool-path generation method for automatic manufacturing of light-frame panels: Steel[2]

3.1 Introduction

3.1.1 Background

Recent advancements in offsite construction methods are accelerating the shift from the traditional, on-site construction approach to offsite prefabrication. In this approach, offsite production facilities accommodate the prefabrication of construction components requiring only final assembly on site. This approach dramatically increases the quality of the final product, labour safety, and project deliverability; furthermore, offsite construction also reduces the impact of labour shortages, poor weather conditions, and material waste. These advantages are especially significant for the Canadian construction industry as low population and cold climate prohibit year-round construction. Moreover, the higher demand for mid-rise residential and commercial buildings requires a material shift from traditional timber to light gauge steel (LGS) structures. Considering that LGS conforms to specified standards, the automated production of LGS panels is a vital step for advancing the philosophy of offsite construction.

Many traditional Canadian homebuilders who have adopted the offsite construction philosophy struggle to implement the required level of automation to gain the benefits of offsite construction. This downside is primarily fueled by the lack of options available for construction machinery, which has traditionally been designed for offshore markets thus

[2] The manuscript appearing as Chapter 3 of this thesis was under review by *Automation in Construction* at the time of publication of this thesis.

making them costly and inflexible for small to mid-size homebuilders. To support the industrialization of the Canadian market, a prototype machine has been designed at the University of Alberta that will assist workers in assembling and securing of light gauge steel framed wall-panels.

The prototype is a semi-automated steel-framing machine that utilizes information from shop drawings to automatically adjust boundaries thereby allowing workers to assemble any applicable LGS panel (Figure 3.1). The completion of manual frame assembly is preceded by the squaring phase, which ensures the quality of the frame as per the shop drawings. After the squaring phase, the soft-connected frame is dragged using front and a back dragging schemes through a stationary gantry consisting of four automatic screw-fastening carriages (SFCs). The dragging embodiments consist of four electromagnetic squares positioned along the corners of the steel frame thereby allowing for proper squaring and synchronized dragging of LGS panels. During the dragging phase, the frame is positioned such that self-drilling screws can be added as required according to the shop drawings. The actual screw-fastening takes place from both ends by means of a custom tool-path algorithm designed to avoid collisions between SFCs and reduce screw-fastening (SF) cycle time. The LGS frame is dragged to each position requiring SF operations, and the SF phase ends after fully securing each connection point. After the completion of the SF phase, the framed wall panel is manually offloaded to the next stage of wall panel construction (Malik et al. 2018). Since the prototype machine is designed for Cartesian operation, any SF operations outside the normal plane must be performed manually prior to the dragging phase.

Recently, extensive efforts have been made to automate construction operations by utilizing various degrees of automation. Bock (2015) highlights a variety of single task construction robots (STCR) and their potential diffusion into built environments through future technological disruptions. Some literature relating to STCR exists, such as steel beam assembly automation (Chu et al. 2013), contour crafting machine (Zhang & Khoshnevis 2013), and Bricklaying Robot (Aguair & Behdinan 2015); however, these studies do not consider the automated wall-framing process, particularly the LGS framing process. The prototype machine, as mentioned above, aims to bridge this gap.

The research describes a novel framework for automating the transfer of manufacturing information from a building information modelling (BIM) output to a steel framing machine prototype (SFMP) using an open-source file format hereafter referred to as a recipe file (.rcp). The aim is to showcase the potential of automated generated shop drawings as proposed by Manrique et al. (2015) and the transfer of such information to motion commands for the programmable logic controller (PLC). The proposed recipe file format is designed for easy extraction of information relating to frame dragging, screw positioning and fastening, and machine adjustment. The proposed framework consists of three unique stages: (1) interpreter; (2) optimizer; and (3) post-processor. The completion of these stages results in safe tool-paths for real-world execution, which is the principal contribution of this research. Here, the function of the interpreter is to extract the data, whereas the optimizer is responsible for generating safe work sequences for various mechanical systems by ensuring proper integration of collision detection and avoidance modules. Subsequently, the post-processor permits the transfer of motion commands to the PLC through the generation of a recipe file.

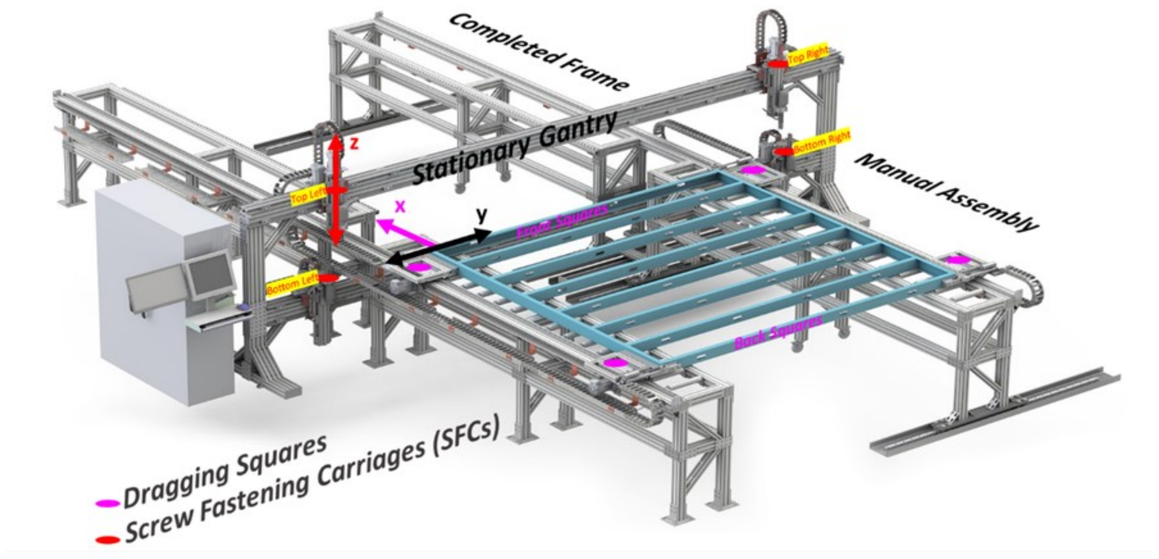


Figure 3.1 Overview of steel framing machine prototype

3.1.2 Literature Review

Typically residential construction uses wood as the primary building material; however, cold-formed steel (CFS) is preferred for commercial and mid-rise residential building projects due to such material properties as its light weight, high durability, strength, and stiffness, combined with consistent quality and abundant supply. These improved material properties also allow for fewer limitations imposed due to seasonal construction (Gupta 2011). Coupled with the increasing demand for commercial and mid-rise residential construction, these characteristics promote the use of LGS. Moreover, innovative systems such as the hybrid LGS wall system with a cellular concrete mix (Gupta 2011) and the prefabricated high-rise steel-frame structure with inclined braces (Liu et al. 2015) aim to further advance the utilization of steel as a major source of building material.

A renewed interest in the construction industry is being fueled by the implementation of modern methods of construction (MMC) that promote tangible benefits through prefabricated construction. Prefabricated construction provides superior quality product as well as improved productivity and sustainability. Lawson and Ogden (2008) highlight various highly prefabricated construction systems and provide case studies relating to offsite manufacturing (OSM) and modular construction. The study promotes a mixed "hybrid" approach of planar and volumetric construction that combines the utilization of long span 2D-panels for open areas and 3D modules for the highly serviced areas, like bathrooms and kitchens. Paudel et al. (2016) discuss prefabricated modular and steel structures and their effectiveness in the structural development process. The study reports an 86% increase in prefabrication and preassembly over the past 15 years, citing MMC as a significant cause of improvements in cost and time performance for several projects. A study by Shahzad, Mbachu, and Domingo (2015) indicates that prefabrication results in a 34% reduction in project time and a 19% reduction in project cost, which translates to a 7% productivity improvement. Their study also reveals that this increase in productivity is not the case for the traditional construction sector, which has been on the decline in recent decades. Bock (2015) argues that the decline in productivity is primarily fueled by low capital investment, resulting in limited R&D, as well as the industry's reluctance in adopting new strategies and technologies; however, the study affirms that the growth of construction automation is inevitable and continued research and development will increase innovation and adoption on a larger scale.

Graph theory is a mathematical formulation that involves a finite non-empty set of vertices that connect with edges. The application of graph theory consists of varying subject matter such as network design, data structure, software engineering, and image processing (Singh 2014). For construction engineering, the network design problems closely resemble various routing problems experienced in real-world operations. More specifically, route problems such as Hamiltonian path, shortest path, and traveling salesman problem are extensively researched for path planning problems. The traveling salesman problem (TSP) is an optimization method where the aim is to find a route through a set of m cities such that each city is visited exactly once with the shortest total traveling distance (Kizilateş & Nuriyeva 2013). The TSP has been studied in applications involving optimization of vehicle routing, computer wiring, wallpaper cutting, and job sequencing. Lawler (1985) published a well-documented book on TSP, which was later revised to include new advancements in combinatorial optimization by Gutin and Punnen in 2006. Unlike TSP, which is NP-complete, the shortest path problem is easier to compute and can be constructed for finding the shortest path between two locations (vertices) using metrics such as cost, distance, and time (Burdett & Kozan 2014).

Adequately addressing TSP helps to facilitate shorter delivery times and leads to increased customer satisfaction, which is critical for the development of successful supply chains. Apart from that pertaining to routing, extensive literature exists on the topic of simulation modelling for supply chain management (Jahangirian et al. 2010). The utilization of simulation modelling allows for in-depth analysis of change over time, which can be modelled through two primary means: (1) continuous modelling, extensively used in the chemical industry (Lee et al. 2002), and (2) discrete-event simulation (DES), most popular technique with extensive use in manufacturing (Jahangirian et al. 2010; Ossimitz & Mrotzek 2008). However, in recent years, discrete-continuous hybrid modelling has been shown to better represent the discrete and continuous nature of supply chain systems (Lee et al. 2002), as well as strategic issues in construction management (Peña-Mora et al. 2008), and operational matters in production systems (Lavoie et al. 2007).

Since the mechanical systems used in this research are well-defined, DES as the preferred method for modelling change over time is selected for this research. The discrete nature of screw-fastening operations allows the simulation to jump from one event to the next without changing the state of the system. This approach of modelling real-world systems has been widely used in manufacturing for understanding manufacturing system designs and operations (Azab & AlGeddawy 2012). Negahban and Smith (2014) surveyed 290 papers on the application of DES in manufacturing and present an increased shift to operations planning and scheduling using DES. Apart from the application of manufacturing systems and operations, many studies apply simulation to improve decision-making in construction. Altaf et al. (2015) present a methodology integrating DES with an automated data acquisition system to examine the real-time production information for enabling practitioners to effectively control production line operations. Another study by Altaf, Al-Hussein, and Yu (2014) develops a DES model that utilizes Particle Swarm Optimization to automate the panel sequencing process for panelized wood construction. Liu et al. (2015) develop a special template for construction practitioners to perform "What-If" analyses of actual production lines for panelized construction. Through mimicking the prefabrication process for an LGS panel production facility, the study, underscores the advantages of the proposed template.

Computer-aided Design (CAD) and Computer-aided Manufacturing (CAM) applications are extensively used in the manufacturing industry, specifically, by means of computer numerical control (CNC) machining. The CAD systems precisely describe models and assemblies as per the real-world, whereas CAM systems convert CAD information into machine understandable language consisting of safe tool-paths for controlling the manufacturing operations (Xu & Newman 2006). However, prior to the actual machine operations, the information generated by CAM systems is further modified by built-in interpreters within CNC machines (Yusof & Latif 2015; Shen et al. 2017). Nevertheless, each CAM system is responsible for generating safe tool-paths, which requires an in-depth analysis of collision detection and avoidance (Zhang & Khoshnevis 2013; Chen et al. 2012; Ren et al. 2010). As the complexity of CNC machining increases, new techniques for collision detection and avoidance using computer vision are proposed to counter the potential collisions with unknown and un-programed obstacles during the actual production (Ahmad & Plapper 2015; Ahmad et al. 2016, 2017).

Providing manufacturing and software solutions, like CNC machining, based on local requirements can further expand the growth and adoption of panelized or modular construction. As more production is undertaken in a factory environment, proposed solutions must be easily adaptable and configurable for the dynamic needs of the construction industry. Extensive research material is available relating to assembly of LGS panels (Allen & Iano 2009; Yu & LaBoube 2010), screw connections (Fiorino et al.

2007), and structural stability (Davies 2006). Nevertheless, at present, few studies are available focus on the design and controls of construction automation equipment (Aguair & Behdinan 2015; Chu et al. 2013; Zhang & Khoshnevis 2013), wherein this limitation is more apparent in the wall fabrication process. However, Tamayo et al. (2017), having investigated a control strategy for a steel framing machine prototype (SFMP), apply their research to the existing prototype design assembled at the University of Alberta, which is the prototype presented in this research. More specifically, due to the absence of literature on automating the process of information transfer between BIM output and automated framing machines, this research aims to provide a novel framework that can be used to safely control Cartesian machines with similar controls.

3.2 System Framework: Mapping for manufacturing residential walls (MMRW)

The proposed framework follows the architecture presented in Figure 3.2, where three key stages are identified as (1) interpreter, (2) optimizer, and (3) post-processor. The interpreter stage allows for data acquisition from existing CAD system outputs and can extract relevant manufacturing information through data parsing. The completion of the interpreter stage outputs requires SF information, frame dimensions, and panel ID. The data input and parsing phase constitutes the interpreter stage, which is inspired by an interpreter proposed by Yusof and Latif (2015).

During the optimizer stage, the filtered manufacturing data is passed through an initial path generation module that outputs four independent tool-paths, each correlating to a given SFC. These primary tool-paths are analyzed using DES wherein time-dependent information for each screwing operation is produced. Finally, the manufacturing data with time domain, obtained from DES, is analyzed for potential collisions between each of the four SFCs and, if required, a collision avoidance module is executed to prevent such events. The conclusion of the optimizer stage ensures the generation of collision-free tool-paths (Ahmad & Plapper 2015; Ahmad et al. 2016; Zhang & Khoshnevis 2013). The modular design for the optimizer stage allows for the universal adoption of the proposed framework with machine parameters, e.g., machine logic and hardware arrangements, only limited to the DES. Here, unlike with the simulation module, both the path generation and collision avoidance modules are widely applicable to Cartesian machines and multiple carriage systems (Zhang & Khoshnevis 2013).

To utilize system-generated manufacturing information, the last stage involves the creation of a recipe file (.rcp) constituting a predetermined format involving motion commands. Here, the open-source format of the recipe file provides options for future expandability. Moreover, the post-processor stage also enables shop floor modifications through an easy to edit comma-separated values (CSV) output.

The proposed framework is programmed using Python (Version 3.6.4) because of its ability to quickly modify numerical data and simulate machine operations through widely available simulation and graphical packages. Moreover, Python open environment architect also allows for powerful and fast programming through its ease of reading, processing, and manipulating information. To implement machine constraints, the MMRW framework is built upon Cartesian movements, which serve as the building block for producing tool-paths required for wall framing. Such Cartesian movements include bidirectional SFC movements along the ±*y*-direction, unidirectional frame transfer along the +*x*-direction, and SF motion along the ±*z*-direction. Since stepper motors generate the Cartesian movements, the framework also allows for dynamic adjustment of motor properties including acceleration, deceleration, and maximum velocity. The following subsections aim to highlight the principal modules contained within each of the three stages, as presented in Figure 3.2.

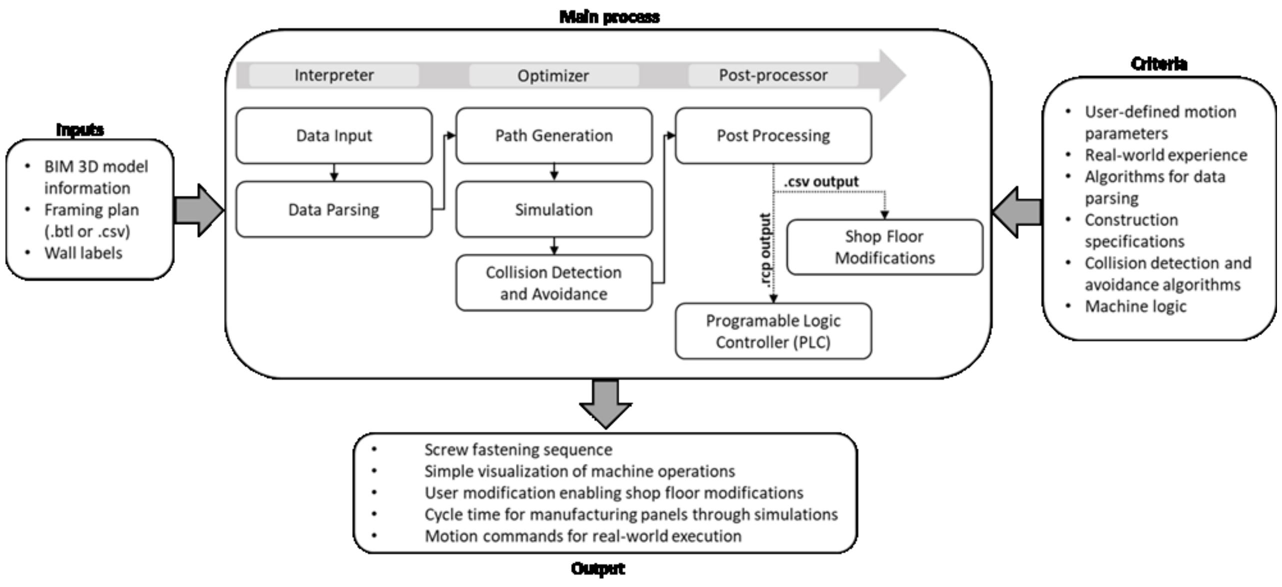

Main process
Interpreter
Optimizer
Post-processor
Inputs
• BIM 3D model information
• Framing plan (.btl or .csv)
• Wall labels
Data Input
Data Parsing
Path Generation
Simulation
Collision Detection and Avoidance
Post Processing
.csv output
.rcp output
Shop Floor Modifications
Programable Logic Controller (PLC)
Criteria
• User-defined motion parameters
• Real-world experience
• Algorithms for data parsing
• Construction specifications
• Collision detection and avoidance algorithms
• Machine logic
• Screw fastening sequence
• Simple visualization of machine operations
• User modification enabling shop floor modifications
• Cycle time for manufacturing panels through simulations
• Motion commands for real-world execution
Output

Figure 3.2 System Framework: Mapping for manufacturing residential walls (MMRW)

3.2.1 Data Input

The data input module allows for importing and reading of either Building Transfer Language (BTL) or CSV files through file browser and data acquisition functions. The file browser function provides a user interface for selecting a file path and allows for importing of raw file data. The imported data is further process through data acquisition function that reads the inputted file (containing m lines) and produces a simplified data array (E_{info}) where each entry (l_i) contains specific operational information relating to machine's operations (Equation 3.2.1.1).

$$E_{info} = \{\forall_i = \{1, \dots, m\},\ l_i \in ID | l_i\ has\ operational\ info\} \qquad (3.2.1.1)$$

where (ID) consists of input data from either BTL or CSV file and "has operational info" condition ensures that each element within E_{info} has screw-fastening information, i.e., the x-, y-, and z-coordinates.

The information relating to the panel's overall dimensions (i.e., L_f = length, H_f = height, and W_f = width) and panel ID (pID) is also stored in separate arrays for the post-processor stage (Figure 3.3). As an initial quality check, Equation 3.2.1.2 compares the panel's overall dimensions with machine's overall operational dimensions (see Figure 3.4) and further steps are only taken if the frame has acceptable dimensions. Upon the completion of the initial quality check, the manufacturing information extracted from either BTL or CSV file is pass to data extraction (parsing) module.

$$\begin{cases} min(H_a) \le H_f \le max(H_a) \\ min(L_a) \le L_f \le max(L_a) \\ min(W_a) \le W_f \le max(W_a) \end{cases} \qquad (3.2.1.2)$$

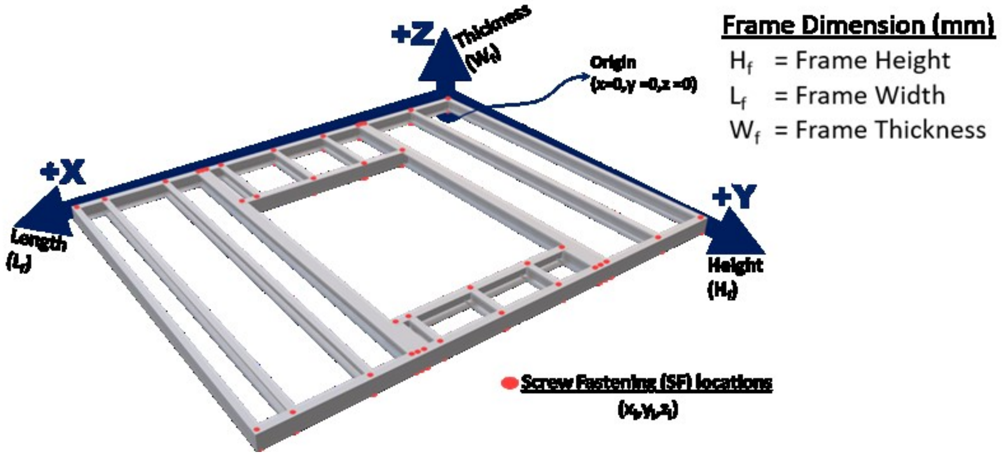


Figure 3.3 Computer-aided model of steel frame with corresponding parameters

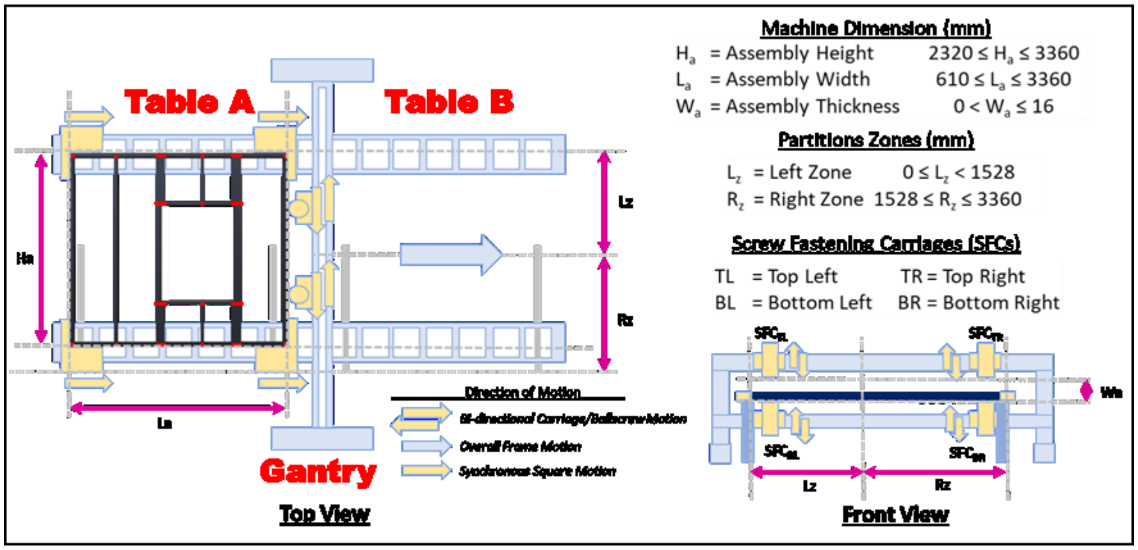


Table A
Table B
Gantry
Top View
Front View
Ha
La
Lz
Rz
Wa
SFC$_{TL}$
SFC$_{TR}$
SFC$_{BL}$
SFC$_{BR}$
Direction of Motion
Bi-directional Carriage/Ballscrew Motion
Overall Frame Motion
Synchronous Square Motion
Machine Dimension (mm)
H_a = Assembly Height $2320 \le H_a \le 3360$
L_a = Assembly Width $610 \le L_a \le 3360$
W_a = Assembly Thickness $0 < W_a \le 16$
Partitions Zones (mm)
L_z = Left Zone $0 \le L_z < 1528$
R_z = Right Zone $1528 \le R_z \le 3360$
Screw Fastening Carriages (SFCs)
TL = Top Left
TR = Top Right
BL = Bottom Left
BR = Bottom Right

Figure 3.4 Steel Framing Machine sketch, including machine parameters and direction of motion

3.2.2 Data Extraction

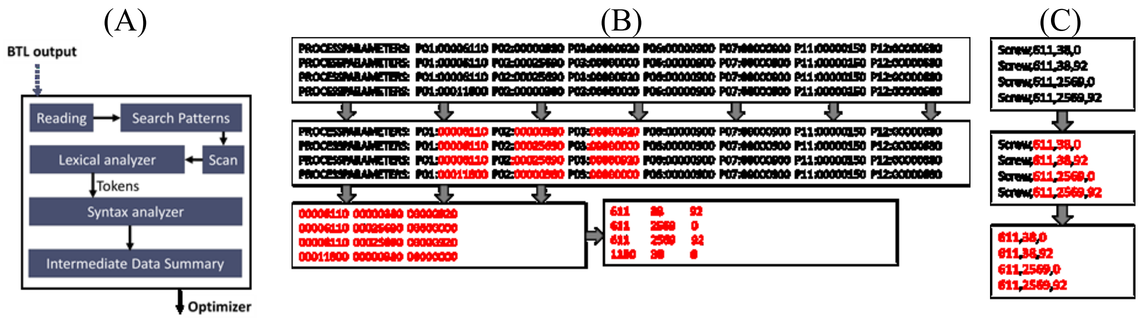
The data extraction module, based on the work of Yusof and Latif (2015), provides data filtering functions that read and simplify SF information (contained within E_{info}) into *x*-, *y*-, and *z*-coordinates (Figure 3.5A). The simplified data from the BTL file is passed through a function that reads and searches for patterns relating to the *x*-, *y*-, and *z*-coordinates (Figure 3.5B). After locating specified patterns, tokenization is conducted by means of scanning each pattern, which converts a sequence of characters into a sequence of tokens. Here, each sequence of tokens is passed through a function that checks for correct syntax followed by the paring of tokens strings to regular expressions relating to *x*-, *y*-, and *z*-coordinates. In the case of a BTL file, each coordinate is further simplified as per the base unit stated in the initial input file. Additional information relating to BTL format is available at design2machine.com. Lastly, all coordinates are validated as per the machine's overall operational dimensions shown in Figure 3.4. The above data extraction technique is repeated for each index within E_{info} array, and the final SF data is stored in a python dictionary (S_{info}) for use in the optimizer phase (Equation 2.2.1).

$$S_{info} = \{\forall_i = \{1, \ldots, size(E_{info})\}, (x_i, y_i, z_i\ ) \in E_{info} | (0 < x_i \leq max(L_a)) \wedge (0 < y_i \leq max(H_a)) \wedge (0 \leq z_i \leq max(W_a))\} \quad (3.2.2.1)$$

where

- x_i (mm) = *x*-position of screw-fastening location, *i* = 1 to *n*
- y_i (mm) = *y*- position of screw-fastening location, *i* = 1 to *n*
- z_i (mm) = *z*-position of screw-fastening location, *i* = 1 to *n*
    - *n* represents the cumulative number of SF operations

The data extraction from a CSV file follows a similar process as described for a BTL file; however, for CSV files the inputted data is structured in an organized manner providing a simplified data extraction process (Figure 3.5C). The data stored in CSV files is assumed to be of similar units.



```
PROCESSPARAMETERS: P01:00006110 P02:00000380 P03:00000920 P06:00000900 P07:00000900 P11:00000150 P12:00000680
PROCESSPARAMETERS: P01:00006110 P02:00025690 P03:00000000 P06:00000900 P07:00000900 P11:00000150 P12:00000680
PROCESSPARAMETERS: P01:00006110 P02:00025690 P03:00000920 P06:00000900 P07:00000900 P11:00000150 P12:00000680
PROCESSPARAMETERS: P01:00011800 P02:00000380 P03:00000000 P06:00000900 P07:00000900 P11:00000150 P12:00000680
```

```
00006110 00000380 00000920
00006110 00025690 00000000
00006110 00025690 00000920
00011800 00000380 00000000
```

```
611    38    92
611    2569  0
611    2569  92
1180   38    0
```

```
Screw,611,38,0
Screw,611,38,92
Screw,611,2569,0
Screw,611,2569,92
```

```
611,38,0
611,38,92
611,2569,0
611,2569,92
```

Figure 3.5 Data parsing subroutine (A); BTL file interpreter process; and CSV file interpreter process (C)

3.2.3 Path Generation (Guided Nearest Neighbour algorithm)

The path generation module produces initial tool-paths for each SFC by grouping the extracted data into four separate categories, i.e., top-right, bottom-right, top-left, and bottom-left (Equation 3.2.3.1). The splitting of extracted data is based on pre-determined machine design, which divides the SF operations into left and right regions, where the left region constitutes any operations requiring a y-position < 1,528 mm and the right side consists of operations requiring a y-position ≥ 1,528 mm (see Figure 3.4). This approach of splitting operational areas allows for the creation of static working and buffer zones for collision avoidance, as explained in Section 3.2.5 above. In short, the inputted manufacturing information is separated into four subsets ($SF_{TR}, SF_{TL}, SF_{BR}, SF_{BL}$), and initial tool-paths are created based on the subsets.

$$\begin{cases} SF_{TR} = \{\forall_i = \{1, \dots, size(S_{info})\}, (x_i, y_i, z_i) \in S_{info} | min(R_z) \leq y_i \leq max(R_z) \wedge z_i > 0\} \\ SF_{TL} = \{\forall_i = \{1, \dots, size(S_{info})\}, (x_i, y_i, z_i) \in S_{info} | min(L_z) \leq y_i < max(L_z) \wedge z_i > 0\} \\ SF_{BR} = \{\forall_i = \{1, \dots, size(S_{info})\}, (x_i, y_i, z_i) \in S_{info} | min(R_z) \leq y_i \leq max(R_z) \wedge z_i = 0\} \\ SF_{BL} = \{\forall_i = \{1, \dots, size(S_{info})\}, (x_i, y_i, z_i) \in S_{info} | min(L_z) \leq y_i < max(L_z) \wedge z_i = 0\} \end{cases} \tag{3.2.3.1}$$

The tool-path generation methodology requires path generation for SFCs, where the cumulative travel distance through each screw location in a given subset is minimized. The minimization of travel distance thus requires path creation without imposing backtracking, as well as optimal point selection to avoid excessive carriage movements. Since the SFMP requires unidirectional frame movement, the path generation methodology can be significantly simplified by adopting the (n + 1)-city TSP as proposed by Lawler (1985). Since the movements of frame and screw carriages are constrained, Nearest Neighbour (NN) algorithm can be applied to solve the (n + 1)-city TSP, which, although it does not guarantee an optimal solution, provides a near to optimal solution. To achieve the abovementioned objectives, a modified NN algorithm based on research presented by Usman (2017) is proposed. Herein referred to as a Guided Nearest Neighbour (GNN), this algorithm requires minimization of tool travel distance as shown in Equation 3.2.3.2. It should be noted that heuristic techniques such as genetic algorithms, particle swarm, artificial bee colony, and ant colony optimization can potentially produce better tool-paths within a reasonable time (Carballo et al. 2017; Gündüz et al. 2015; Rui 2017); however, such algorithms require the integration of machine logic, which would severely restrict the applicability of the present research. Here, GNN provides a deterministic approach for generating tool-paths and does not require any information about mechanical systems, which can be explored separately (see Section 3.2.4 and 3.2.5). The distance minimization is constrained by Equation 3.2.3.3, which restricts both the number of approaches and departures of a screwdriver, to a given screw-fastening location to one, whereas Equation 3.2.3.4 adds binary variables to different connections ($Q_{i,j}$) between each screw-fastening operation. The final constraint to avoid backtracking is applied through a systematic procedure as detailed below.

$$Minimize \sum_{i=1}^{n} \sum_{j=1}^{n} d_{i,j} Q_{i,j} \tag{3.2.3.2}$$

$$\sum_{j=1}^{n} Q_{i,j} = 2 \qquad \forall_i \in V \tag{3.2.3.3}$$

$$Q_{i,j} = \begin{cases} 1 & \text{if edge (i, j) is in final path} \\ 0 & \text{else.} \end{cases} \tag{3.2.3.4}$$

where

- $Q_{i,j}$ = Binary condition for edge connecting vertex i and j
- $d_{i,j}$= Distance between each pair of vertices i and j

Prior to beginning SF operations, each carriage is positioned beyond the right and left frame edges (ST_{SFC}). Therefore, the proposed GNN algorithm allows each carriage to start with the nearest vertex to its respective position, as presented in Figure 3.6. To simplify the path selection, sub-domains (DP_k) are created for each unique x-value, which allows for path creation at a given frame position (see Equation 3.2.3.5). The distance between each element within DP_k and the current vertex (CV) is calculated using Equation 3.2.3.6. Initially, CV is set to ST_{SFC} and, using Equation 3.2.3.8, a new CV value is calculated that represents the first (nearest) vertex. This vertex in turn is added to a tool path (TP_{SFC}) array representing the first operation for a given SFC.

$$DP_k = \{\forall_i = \{1, \dots, n\},\ (x_i, y_i, z_i) \in SF_{SFC} | x_i = x_k \in RO_{SFC}\} \tag{3.2.3.5}$$

$$disA = \left\{\forall_i = \{1, \dots, size(DP_k)\}, \left\{x_j, y_j, z_j, d_{CV,j}\right\} \in DP_k\right\} \tag{3.2.3.6}$$

$$d_{i,j} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2} \tag{3.2.3.7}$$

$$CV = \left\{\forall_i = \{1, \dots, size(disA)\},\ (disA_i^1, disA_i^2, disA_i^3) \in disA | disA_i^4{}_i = min\left(disA_j^4 \forall_i = \{1, \dots, size(disA)\}\right)\right)\right\} \tag{3.2.3.8}$$

where

- DP_k = Sub-domain containing SF information with similar x-values, where k = 1,...,size(RO_{SFC})
- disA = Array containing vertices information and distance between each listed vertex
- $d_{CV,j}$ = Distance from current vertex (CV) to vertex j
- CV = location of current vertex
- TP_{SFC} = Array containing ordered tool path for a given carriage
- RO_{SFC} = Ascending ordered list of unique x-values from SO_{SFC}
    - SFC = {TR, TL, BL, BR}

After the first vertex has been determined, the DP_k array is updated to exclude CV, while a combination of Equation 3.2.3.6 and 3.2.3.8 is used to calculate the distance to each of the remaining vertices at the given frame position (i.e., x-position represented by index k). Here, each iteration outputs a new CV value, which, in turn, reduces and increases the size of the DP_k and TP_{SFC} arrays, respectively. This process allows the carriage to select the next nearest vertex to its current vertex, thus enabling mapping of the remaining operations in a predetermined manner. The sequence continues until no operations remain at the specified frame position.

Considering that SFCs have a range of y-values, at this stage any given carriage can travel in a bi-directional fashion (Figure 3.6, see "bi-directional edge"). Nevertheless, due to the unidirectional nature of the frame movement, the next vertex must not only provide the shortest edge connecting the current vertex and unvisited vertices, but it must also avoid excessive travel. In other words, selecting a middle vertex at a subsequent frame position will necessitate backtracking to visit the remaining unvisited vertices within the subsequent frame position (Figure 3.6, see “suboptimal edge”). To avoid such backtracking, the next vertex at subsequent frame position is chosen from the outer-most vertices, wherein the smallest distance between the current vertex and the external-most vertices is adopted as the next vertex. Here, using Equation 3.2.3.10, the outer-most vertices are stored in a transitional sub-domain (DT_{k+1}), which, depending on the number of vertices at subsequent frame positions, can contain at most 2 elements. Upon updating DT_{k+1}, as with Equation 3.2.3.6, Equation 3.2.3.11 is used to calculate the distances, followed by generation of a new CV and subsequent incrementation of TP_{SFC}. Upon generation of the new CV value, the *x*-value from the new CV is used to update Equation 3.2.3.5 (excluding the inclusion of said CV), which, in turn, allows for the repeat execution of Equations 3.2.3.6 and 3.2.3.8, thereby reducing and increasing the size of DP_k and TP_{SFC} arrays, respectively.

$$DI_{k+1} = \{\forall_i = \{1, \dots, n\},\ (x_i, y_i) \in SF_{SFC} | x_i = x_{k+1} \in RO_{SFC}\} \tag{3.2.3.9}$$

$$DT_{k+1} = \begin{cases} (size(DI_{k+1}) = 1) \to \{(x_{k+1}, min(DI_{k+1}^2))\} \in DI_{k+1} \\ (size(DI_{k+1}) > 1) \to \{(x_{k+1}, min(DI_{k+1}^2)), (x_{k+1}, max(DI_{k+1}^2))\} \in DI_{k+1} \end{cases} \tag{3.2.3.10}$$

$$\mathrm{disA} = \left\{\forall_\mathrm{i} = \{1,2\}, \left\{\mathrm{x_j}, \mathrm{y_j}, \mathrm{z_j}, \mathrm{d_{CV,j}}\right\} \in \mathrm{DT_{k+1}}\right\} \tag{3.2.3.11}$$

where

- DI_{k+1} = intermediate sub-domain containing operations relating to *x*-position (RO_{SFC}^{k+1})

- DT_{k+1} = transitional sub-domain containing outer-most vertices from DI_{k+1} | $DT_{k+1} \subseteq DT_{k+1}$

The above approach guarantees, that for each new *x*-value, the carriage moves to either a maximum or a minimum *y*-value, thereby permitting the carriage to complete the remaining operations without backtracking. In short, operations are selected in such a manner that each carriage always travels to its nearest neighbouring vertex, with an exception that excludes central points during the transition to a different *x*-value.

A simplified pseudo-code and graphical representation of the proposed algorithm can be found below.

1. *Set sub-domain (DP_k) to only include vertices with same x-values as the vertex closest to ST_{SFC}. Set ST_{SFC} to current vertex;*
2. *Calculate the shortest edge connecting current vertex and an unvisited vertex V;*
3. *Set current vertex to V and NP to vertex coordinates;*
4. *Mark V as visited and set edge ($Q_{i,j}$) to 1, including appending NP to TP_{SFC};*
5. *If all the vertices in the sub-domain are visited, then move to step 7, otherwise repeat from step 2;*
6. *If all the vertices in the domain (DP_k) are visited, then terminate and output the sequence of the visited vertices;*

7. *Set sub-domain2 (DT_{k+1}) to only include vertices representing the maximum and minimum* y*-values from the next set of* x*-values (DI_{k+1});*

8. *Calculate the shortest edge connecting current vertex and an unvisited vertex V from sub-domain2;*

9. *Set current vertex to V and NP to vertex coordinates;*

10. *Mark V as visited and set edge ($Q_{i,j}$) to 1, including appending NP to TP_{SFC};*

11. *Repeat from step 3.*

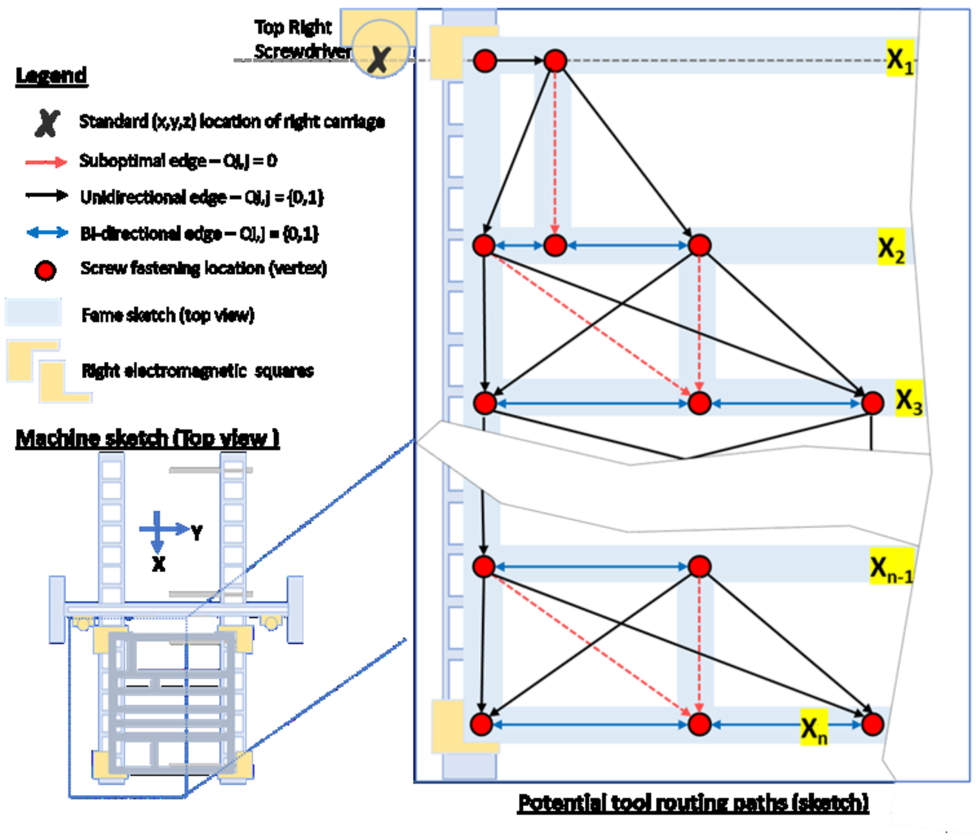

Top Right Screwdriver
Legend
Standard (x,y,z) location of right carriage
Suboptimal edge – Qi,j = 0
Unidirectional edge – Qi,j = {0,1}
Bi-directional edge – Qi,j = {0,1}
Screw fastening location (vertex)
Fame sketch (top view)
Right electromagnetic squares
Machine sketch (Top view)
Y
X
X1
X2
X3
Xn-1
Xn
Potential tool routing paths (sketch)

Figure 3.6 Guided Nearest Neighbour (GNN) algorithm for screw-fastening

The path generation module outputs an independent path for each carriage, which results in the initial paths being inclusive to their respective operations. However, since all carriages are mounted on the stationary gantry, initial tool-paths must be analyzed to understand the potential interdependencies among the carriages.

3.2.4 Simulation: Integration of schedule (time) information

The path simulation module is designed to simulate the tool-paths using DES wherein the basic machine logic mimics the real-world system. The execution of tool-paths in DES allows for intelligently linking 3D SF locations with schedule-related information. The schedule (time) information is critical for predicting possible collisions between SFCs, which in turn allows for the development of collision avoidance systems. The creation of DES also helps predict the cycle times for the actual wall-framing phase, which can be used for the scheduling sequence of panel production.

The operational logic for SF begins after the manual assembly of an LGS panel, where hard connections comprising metal self-drilling screws are used to secure the components of the LGS frame. More specifically, as presented in Figure 3.7, the completion of the manual step is preceded by sequential execution of motion commands as declared by tool-path generation. Initially, synchronized frame dragging and positioning of SFCs takes place in parallel such that the frame moves in the +*x*-direction, and the carriages are positioned at required *y*-positions for the upcoming SF operations. After the proper positioning of the frame, hard connections are made using SFCs positioned above and below the LGS frame.

Figure 3.7 provides a brief overview of machine logic, from the reading of recipe data to the executing of motion commands for various machine components. For this module, the DES model is constructed using Simpy, which is a process-based DES framework for Python programming language. In the Simpy environment, each activity, such as frame and carriage movements, are encompassed in separate processes with eight resources consisting of four carriages and four ball screws. Moreover, the SF data for the simulation model is transferred in discrete packets that include the SF information at a given *x*-position and the first SF location for the upcoming *x*-value (represented as A in Figure 3.7). These discrete data packets ensure that the carriages only operate on screw coordinates at the given frame position, whereas the last data entry allows for the carriages to move to the next operation directly after completing their required SF operations at the current location. The movement of carriages to the next *y*-position after completing SF operations at a given *x*-value ensures that all non-utilized carriages are moved such that the carriages are in place for the next operation as soon as the frame is positioned at the required *x*-position.

It should be noted that the right and left SFCs work independently, whereas the SFCs on the same side operate in a synchronized fashion such that the top SF operations are always conducted before the execution of bottom SF operations (represented as B in Figure 3.7). Moreover, the top screwdriver remains fully extended during the bottom SF operation, and after the completion of bottom SF operations, each screwdriver retracts to an operational position. After the completion of SF operations at a given frame position, the frame is repositioned at the next SF position, and the sequence repeats until all operations have been completed (represented as C in Figure 3.7). As stated previously, SFCs work continuously such that upon completion of one operation they move to the next *y*-position in preparation for the steel frame to be repositioned.

Using the above simulation model each SF coordinate can be modified to include schedule information describing the time of arrival and departure of a given SFC. Here, the time of arrival is calculated using kinematic equations (Equation 3.2.4.1 and 3.2.4.2), where using motor parameters the distance can be converted to an incremental time required for repositioning any given SFC. Furthermore, the departure time is recorded when an SFC departs from a given operation. In short, the simulation model inputs 3D-coordinate information for each SFC and outputs the respective time of execution for each screw. For more natural processing, the embedded arrays containing SF information (Equation 3.2.3.1) are extended with time-related information as below:

$$[[x_1, y_1, z_1, t_{A1}, t_{D1}], [x_2, y_2, z_2, t_{A2}, t_{D2}] \ldots, [x_n, y_n, z_n, t_{An}, t_{Dn}]]$$

where

- t_{Ai} = time of approach for i = 1 to n (sec)
- t_{Di} = time of departure for i = 1 to n (sec)

  *n represents the cumulative number of SF operations for a given SFC

$$d = v_i t + \frac{1}{2} a t^2 \qquad (3.2.4.1)$$

$$v_f^2 = v_i^2 + 2ad \qquad (3.2.4.2)$$

where

- v_i = initial velocity (mm/s)

- v_f = final velocity (mm/s)
- a = acceleration (mm/s^2)
- d = distance (mm)
- t = time (s)

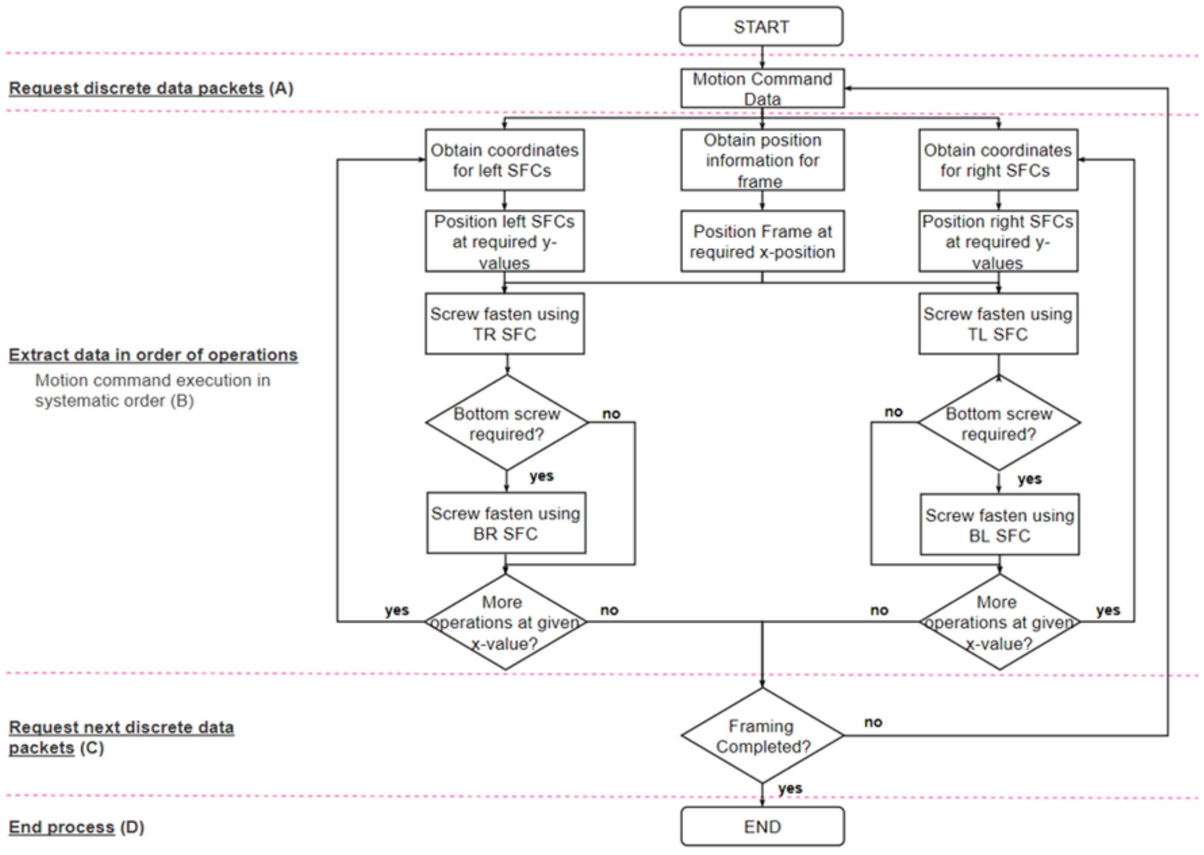

START
Request discrete data packets (A)
Motion Command Data
Obtain coordinates for left SFCs
Obtain position information for frame
Obtain coordinates for right SFCs
Position left SFCs at required y-values
Position Frame at required x-position
Position right SFCs at required y-values
Screw fasten using TR SFC
Screw fasten using TL SFC
Extract data in order of operations
Motion command execution in systematic order (B)
Bottom screw required?
no
yes
Screw fasten using BR SFC
Screw fasten using BL SFC
More operations at given x-value?
Request next discrete data packets (C)
Framing Completed?
End process (D)
END

Figure 3.7 Screw-fastening logic for steel framing machine prototype (SFMP)

Here, using Equation 3.2.4.2 and assuming zero initial velocity, the $d_{threshold}$ can be found using Equation 3.2.4.3, which states the minimum distance to achieve the maximum velocity. Depending on manufacturing information, each carriage or frame positioning requires a specified travel distance (d_{move}), which can either be greater than or less than $d_{threshold}$. For cases in which the distance is smaller than $d_{threshold}$, the linear motion device will not experience v_{max} (case 2), whereas the opposite will be true if the distance is greater than $d_{threshold}$ (case 1). The distinction between either of the above two cases is critical for predicting the travel time for various discrete movements, wherein incorrect assumptions for time will result in incorrect scheduling of SF operations, thereby potentially instigating carriage collision during actual SF operations. Using basic kinematic equations, the approximate time required to move various distances can be found as expressed in Equations 3.2.4.4 and 3.2.4.6 representing cases 1 and 2, respectively.

$$d_{\text{threshold}} = \frac{{v_{max}}^2}{2a_i} \tag{3.2.4.3}$$

<u>Case 1: $d_{move} \geq d_{threshold}$</u>

$$t_f = t_a + t_{max} + t_d$$

$$t_f = \frac{v_{max}}{a_i} + \frac{d_m}{v_{max}} + \frac{v_{max}}{a_d} \tag{3.2.4.4}$$

<u>Case 2: $d_{move} < d_{threshold}$</u>

$$t_f = t_a + t_d$$

$$v_a = \sqrt{\frac{2\text{d}_{\text{move}}\, a_a a_d}{a_a + a_d}} \tag{3.2.4.5}$$

$$t_f = \frac{v_a}{a_i} + \frac{v_a}{a_d} \tag{3.2.4.6}$$

where

- d_{move} = distance to move (mm)
- d_m = distance travel at V_{max} (mm)

- $d_{threshold}$ = distance to achieve maximum velocity (mm)
- a_i = acceleration (mm/s^2)
- a_d = deceleration (mm/s^2)
- v_{max} = maximum velocity (mm/s)
- v_a = velocity achieved (mm/s)
- t_{max} = time require at v_{max} (s)
- t_a = time for acceleration (s)
- t_d = time for deceleration (s)
- t_f = final time to move d_{move} (s)

3.2.5 Collision Detection and Avoidance

The collision detection and avoidance module serves the function of collision prevention between the left and right pairs of SFCs. The prototype machine design allows for the creation of four unique zones consisting of two buffers and two working zones (Figure 3.8). The buffer zones are set up near the shared border between the right and left working zones to prevent collision between the carriages. The size of each buffer zone must be greater than the width of the SFCs to avoid potential collisions. Moreover, the creation of such buffer zones ensures that carriages will never collide when they are operating outside of the buffer zones (Zhang & Khoshnevis 2013). For a collision to occur the following conditions must be met: (1) each buffer zone requires SF operations, and (2) the operations are planned at concurrent times. In short, the clash between carriages can be avoided by scheduling the operations such that two carriages on the same linear guide are never operating in buffer zones at similar timeframes.

To check for imminent collisions between carriages the module performs collision detection by analyzing screw locations and the respective approach and departure times assigned to each carriage. Since carriages can relocate to the next SF location while waiting for the frame to be moved into the correct position, collisions must be prevented during the SF operations and while idling for frame relocation. Therefore, to avoid collisions between two carriages, the following must be true:

$$\mathrm{Abs}(y_1(t)-y_2(t)) > \text{Carriage Width} = 340\ \text{mm} \tag{3.2.5.1}$$

where $0 < t <$ time of the SF phase; and $y_1(t)$ and $y_2(t)$ represent the *y*-position of the two carriages on the same linear guide at time, t.

More specifically, for a given SFC, each operation in the buffer zone can be isolated using Equation 3.2.5.2 and, considering Equation 3.2.5.1, Equation 3.2.5.3 combines top and bottom operations in separate arrays, thus allowing for generation of collision points via Equation 3.2.5.4 and 3.2.5.5.

$$\begin{cases} \mathrm{BP_{TR}} = \{\forall_i = \{1,\dots,\mathrm{size}(\mathrm{TP_{TR}})\}, (x_i, y_i, z_i, t_{Ai}, t_{Di}) \in \mathrm{TP_{TR}} \mid max(R_z) \le y_i \le max(R_z) + CW \ \wedge z_i > 0\} \\ \mathrm{BP_{TL}} = \{\forall_i = \{1,\dots,\mathrm{size}(\mathrm{TP_{TL}})\}, (x_i, y_i, z_i, t_{Ai}, t_{Di}) \in \mathrm{TP_{TL}} \mid \max(L_z) - \mathrm{CW} \le y_i < \max(L_z) \wedge z_i > 0\} \\ \mathrm{BP_{BR}} = \{\forall_i = \{1,\dots,\mathrm{size}(\mathrm{TP_{BR}})\}, (x_i, y_i, z_i, t_{Ai}, t_{Di}) \in \mathrm{TP_{BR}} \mid max(R_z) \le y_i \le max(R_z) + CW \ \wedge z_i = 0\} \\ \mathrm{BP_{BL}} = \{\forall_i = \{1,\dots,\mathrm{size}(\mathrm{TP_{BL}})\}, (x_i, y_i, z_i, t_{Ai}, t_{Di}) \in \mathrm{TP_{BL}} \mid \max(L_z) - \mathrm{CW} \le y_i < \max(L_z) \wedge z_i = 0\} \end{cases}$$

(3.2.5.2)

$$\begin{cases} \mathrm{BP_{Top}} = \mathrm{CP_{TR}} \cup \mathrm{CP_{TL}} \\ \mathrm{BP_{Bottom}} = \mathrm{CP_{BR}} \cup \mathrm{CP_{BL}} \end{cases} \tag{3.2.5.3}$$

$$CP_{top} = \{\forall i, j \in \{1, \dots, size(BP_{Top})\},\ j \ne i, x_i \in BP_{Top} \mid x_i = x_j, (\max(0, \min(t_{Ai}, t_{Di}) - \max(t_{Aj}, t_{Dj}) + 1) > 0\}$$

(3.2.5.4)

$$CP_{bottom} = \{\forall i, j \in \{1, \ldots, size(BP_{bottom})\}, j \neq i, x_i \in BP_{bottom} | x_i{=}x_j, (\max(0, \min(t_{Ai}, t_{Di}) - \max(t_{Aj}, t_{Dj}) + 1) > 0\}$$

(3.2.5.5)

where

- BP_{SFC} = Operations within buffer zone for a given SFC, such that $BP_{TR} \subseteq TP_{TR}$, $BP_{TL} \subseteq TP_{TL}$, $BP_{BR} \subseteq TP_{BR}$, and $BP_{BL} \subseteq TP_{BL}$
- BP_{Top} = Operations within both top buffer zones
- BP_{Bottom} = Operations within both top buffer zones
- CP_{top} = x-values relating to potential collision events between top carriages
- CP_{bottom} = x-values relating to potential collision events between bottom carriages

The following simplified pseudo-code checks if the y-distance is less than 340 mm between two carriages at any given time (t).

```
1. Get all SF locations (vertices) in the buffer zones;
2. Select and compare the distance between two pairs of vertices from different zones;
3. For vertices that have distance less than 340 mm, check the time of arrival and departure;
4. If there exists no overlap between arrival and departure times, mark that pair of vertices as checked;
```

5. *If there exists an overlap, record the* x*-location in a separate array, mark that pair of vertices as checked;*
6. *Repeat from step 2 until all pairs of vertices are analyzed.*

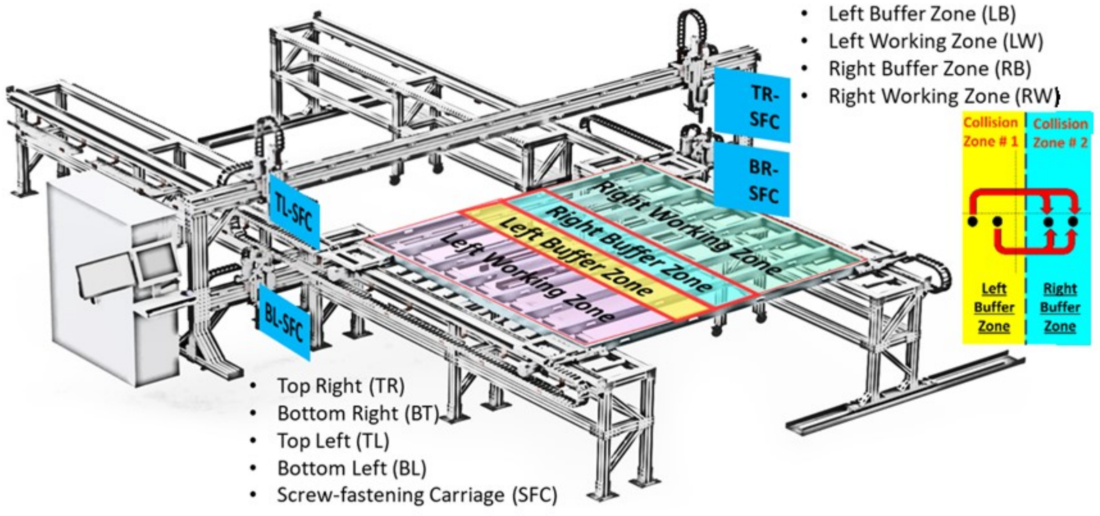


Figure 3.8 Steel framing machine working and buffer zones

In the case of collisions, the program outlined above outputs *x*-locations (CP_{top} and CP_{bottom}) corresponding to such events. The summary of the given locations is passed on to the collision avoidance function module, which intelligently modifies the tool-paths according to the six predetermined strategies and outputs the final tool-paths that provide the least cycle time. The predetermined strategies involve either changing one or both tool-paths as presented in Figure 3.9 under "Collision Avoidance Solutions". These strategies are designed to modify tool-paths by rescheduling the order of operations, which thus affects the time of arrival and departure. Collision avoidance solutions can be explained using one of three options: (1) no change; (2) modify; and (3) update. Here, the "no change" option maintains the original tool-path for a given SFC, whereas the "modify" option changes the tool-path to prevent potential collisions at a given *x*-position. For instance, in strategy 1, the "modify" option allows the right carriage to clear its buffer zone before the arrival of the second carriage to its respective buffer zone. Correspondingly, in strategy 3, the "modify" option allows the left carriage to clear its buffer zone before the arrival of the second carriage to its respective buffer zone. Due to this modification, all carriages travel in a parallel fashion, which consequently ensures safe operations. For strategy 2, tool-paths are modified such that the left carriage starts from its working zone and the right carriage starts from its buffer zone. Again, such a change allows both carriages to work in parallel without the risk of collision.

The "update" option applies "modify" methodology followed by reassessment of the remaining tool-path using the GNN algorithm. This approach allows for the regeneration of the tool-path starting from the end-point of the modified tool-path. More specifically, since the "modify" operation changes the original tool-path, the "update" option allows for the reassessment of remaining operations. Therefore, strategies 4, 5, and 6 are the extension of strategy 1, 2, and 3, respectively. The collision avoidance methodology executes in the manner of piece-wise steps, where each collision point is resolved before investigating the remaining collision events. Since each path is modified independently, path simulation is conducted after each modification in order to understand the interactions among various operations.

Lastly, if the collision between SFCs, at a specified *x*-location, persists after the modification of tool-paths, the strategy is deemed ineffective and is overlooked. All six strategies are executed in sequence, and successful solutions are ranked in ascending order based on their respective cycle times. If successfully applied, each strategy is designed to reliably prevent collisions; however, all six strategies have a high probability of failure if the time in the right working zone is greater than or equal to the time in the left buffer zone, and the time in the left working zone is greater than or equal to the time in right buffer zone. When such an event occurs, the user (operator) is summoned to provide a manual solution, which requires re-uploading of the CSV file with modified coordinates. Following the completion of necessary processes, the final tool-paths are chosen based on the shortest cycle time followed by the rank of the given strategy.

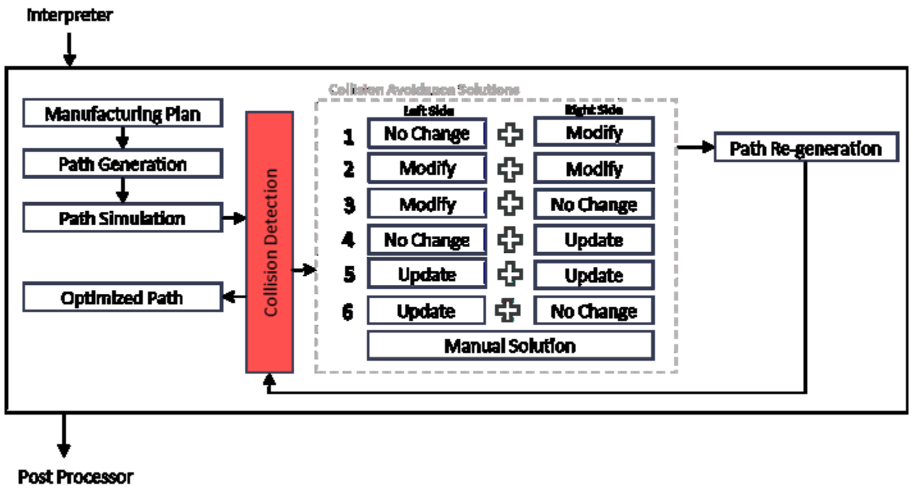


Figure 3.9 Collision avoidance module

3.2.6 Post-processing

The post-processing module is designed to output a recipe file that can be interpreted by the PLC as motion commands to operate the SFMP. This module also generates a CSV file that outputs manufacturing information in a user-friendly manner allowing for in-shop modifications. Here, the user-modified file can be fed back into the interpreter for the generation of a modified recipe file.

The module for generating the final recipe file consists of four primary functions comprising data acquisition, path sorting, data formatting, and output file formatting. The data acquisition function gathers required information relating to frame dimensions (L_f, H_f, and W_f), number of operations, frame ID (*pID*), and optimized tool-paths. This information is passed to the path-sorting function that organizes data such that all the information relating to a given SFC, at a *x*-position, is outputted in a single line starting with the *x*-position and followed by one or more pairs of *y*- and *z*-positions (see Figure 3.10, “Tool Path commands”). This formatting allows the PLC to read the recipe file and quickly extract the information for each tool based on its tool name. Since the frame is moved to discrete positions, the recipe file also prints information relating to each dragging position. The path-sorting information is passed to the data formatting function that formats the information as presented in Figure 3.10. Finally, the sorted information is outputted as a recipe (.rcp) file, which is uploaded to the PLC for final execution.

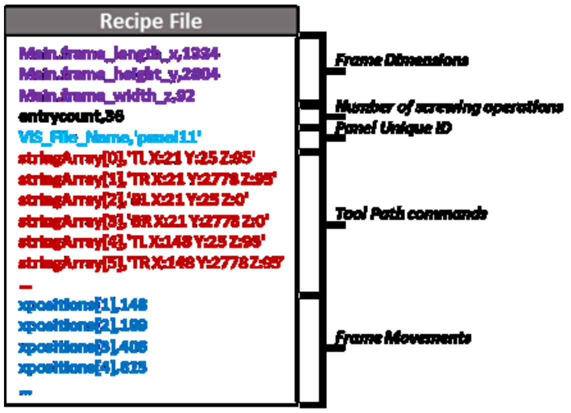
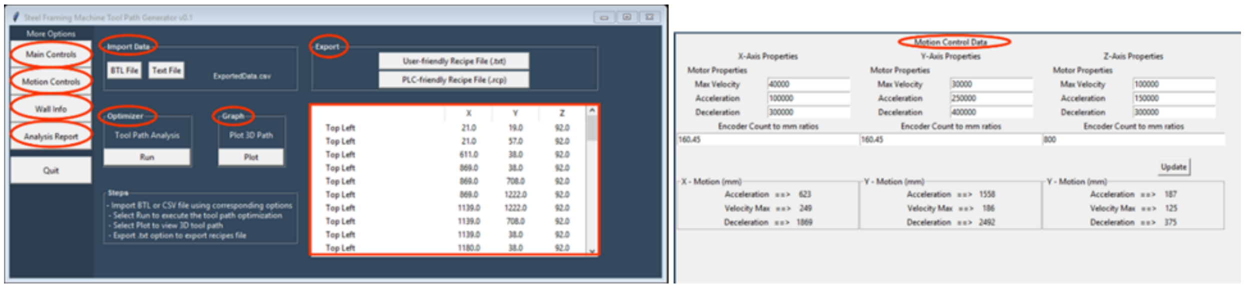

```
Recipe File
Main.frame_length_x,1334
Main.frame_height_y,2804
Main.frame_width_z,92
entrycount,36
VIS_File_Name,'panel11'
stringArray[0],'TL X:21 Y:25 Z:95'
stringArray[1],'TR X:21 Y:2778 Z:95'
stringArray[2],'BL X:21 Y:25 Z:0'
stringArray[3],'BR X:21 Y:2778 Z:0'
stringArray[4],'TL X:148 Y:25 Z:95'
stringArray[5],'TR X:148 Y:2778 Z:95'
...
xpositions[1],148
xpositions[2],189
xpositions[3],408
xpositions[4],825
...
```

Figure 3.10 File structure of recipe file

3.2.7 GUI Design

The graphical user interface (GUI) for the MMRW framework consists of four tabs: main controls, motion controls, wall info, and analysis report as presented in Figure 3.11. The main controls tab includes options for uploading CSV or BTL files, which is followed by the execution of the optimizer that generates a static table with proposed tool-paths and allows for the ability to view the data in a graphical manner using the "Graph" option. Subsequently, export options can output a recipe file for execution on the SFMP and CSV file for shop modifications. The motion controls tab displays the acceleration, deceleration, and maximum velocity values for each Cartesian direction. Finally, wall info and analysis report tabs include the information for frame dimensions and the summary of optimizer output including cycle times and analysis of various collision prevention solutions.



Steps
- Import BTL or CSV file using corresponding options
- Select Run to execute the tool path optimization
- Select Plot to view 3D tool path
- Export .txt option to export recipes file

	X	Y	Z
Top Left	21.0	19.0	92.0
Top Left	21.0	57.0	92.0
Top Left	611.0	38.0	92.0
Top Left	869.0	38.0	92.0
Top Left	869.0	708.0	92.0
Top Left	869.0	1222.0	92.0
Top Left	1139.0	1222.0	92.0
Top Left	1139.0	708.0	92.0
Top Left	1139.0	38.0	92.0
Top Left	1180.0	38.0	92.0

X-Axis Properties		Y-Axis Properties		Z-Axis Properties	
Motor Properties		Motor Properties		Motor Properties	
Max Velocity	40000	Max Velocity	30000	Max Velocity	100000
Acceleration	100000	Acceleration	250000	Acceleration	150000
Deceleration	300000	Deceleration	400000	Deceleration	300000
Encoder Count to mm ratios	160.45	Encoder Count to mm ratios	160.45	Encoder Count to mm ratios	800

X - Motion (mm)		Y - Motion (mm)		Y - Motion (mm)	
Acceleration ==>	623	Acceleration ==>	1558	Acceleration ==>	187
Velocity Max ==>	249	Velocity Max ==>	186	Velocity Max ==>	125
Deceleration ==>	1869	Deceleration ==>	2492	Deceleration ==>	375

Figure 3.11 Graphical user interface (GUI) for the proposed framework: main window (right) and motion control datasheet (left)

3.3 Validation results: Case Studies

Several design cases based on a prefabricated home plan are studied to validate the applicability of the proposed framework. The cycle times and final product from the execution of motion commands are compared with simulated cycle times and 3D-BIM models, respectively. Furthermore, the motion parameters are presented in Table 3.1, where the time for frame dragging and lateral carriage movements is predicted using kinematic equations (as stated in Section 3.2.4) and the ball screw cycle times are assumed to be constant. Here, the difference in left and right ball screw cycle times is the result of the various mounting configurations of mechanical systems. The case studies are analyzed using a Windows 10 computer with Intel Core i7-4790 CPU, and 16 GB RAM.

Table 3.1 Motor parameters used for generating recipe files and for operating steel framing machine (*160.45 Encoder = 1 mm)

Direction	Acceleration (Encoder* /sec^2)	Deceleration (Encoder /sec^2)	Max. Velocity (Encoder /sec)	Cycle Time (sec)
X – Frame Dragging	100,000	300,000	40,000	Variable
Y – Lateral Carriage Movements	150,000	300,000	30,000	Variable
Z – Ball screw Cycle Time (Left)	N/A	N/A	N/A	4.24 sec – Average
Z – Ball screw Cycle Time (Right)	N/A	N/A	N/A	5.03 sec – Average

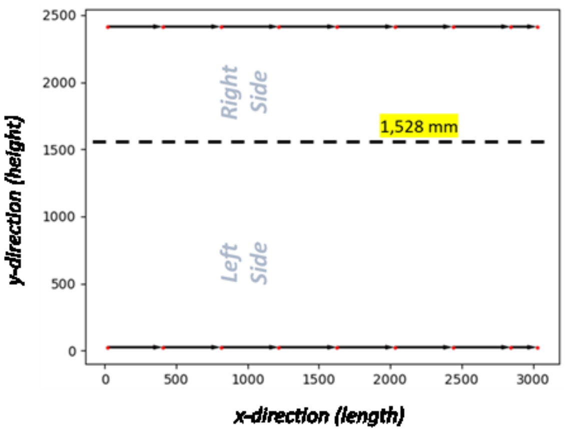
3.4.1 Case Study 1

For the first case study, the production of a normal wall panel is investigated, which includes two tracks and multiple parallel studs positioned 16 in apart. This configuration limits the SF operations to the top and bottom tracks. A brief description of the wall panel is presented in Figure 3.12A, where a total of 36 SF operations are required with a predicted cycle time of 1 minute. Considering no lateral bracings or components are present, the generated tool-paths execute the SF operations in ascending order as presented in Figure 3.12B. Furthermore, since all operations are limited to the frame edges, the tool-paths illustrated in Figure 3.12B uniquely describe the frame movements as required for SF positions. During the actual machine operation, the recipe file (Figure 3.12C) is uploaded to PLC and requires one minute and three seconds to execute all motion commands. The cumulative time for generating the required recipe file is less than one second.

(A)

Shop Drawing	
No. of Operations	36
Frame Dimensions (mm)	Length (x) = 3,048 Height (y) = 2,439 Width (z) = 92
Collision Detection	N/A
Predicted Time (min:sec)	01:00
Actual Time (min:sec)	01:03

(B)



(C)

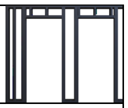
```
Recipe File
Main.frame_length_x,3048
Main.frame_height_y,2439
Main.frame_width_z,92
entrycount,36
VIS_File_Name,'panel1'
stringArray[0],'TL X:21 Y:25 Z:95'
stringArray[1],'TR X:21 Y:2413 Z:95'
stringArray[2],'BL X:21 Y:25 Z:0'
stringArray[3],'BR X:21 Y:2413 Z:0'
stringArray[4],'TL X:406 Y:25 Z:95'
stringArray[5],'TR X:406 Y:2413 Z:95'
…
```

Figure 3.12 Case study 1 summary table (A); top-down tool-path output; and recipe file output excerpt (C)

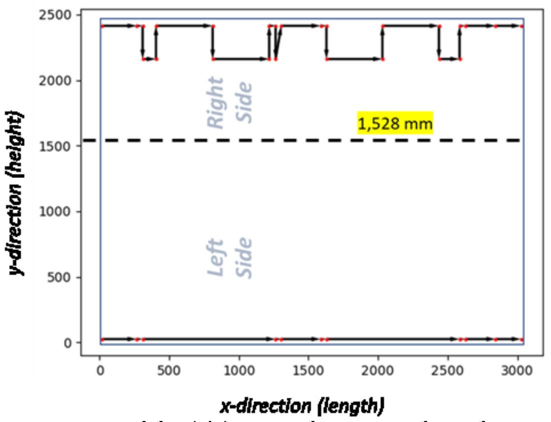
3.4.2 Case Study 2

For the second case study, the production of a wall panel with a cutout for two adjacent doors is investigated. The stud spacing remains at 16 in, whereas the addition of two doors adds five cripple studs with two headers and four king studs. These additional structural changes result in more operations for the right SFCs, including five cases in which the left SFCs require no action due to the addition of cripple studs. As in case study 1, no operations are required within the buffer zones thereby allowing for simple maneuvering of SF operations as illustrated in Figure 3.13B. The illustrated tool-paths show two different paths that exhibit the complex and straightforward nature of the right and left side, respectively. Here, changes in the *x*-direction are defined by frame dragging, whereas changes in *y*-positions are indicative of SFC motion. Nevertheless, these two actions are combined in Figure 3.13B to showcase the overall order of SF operations. A brief description of the panel is presented in Figure 3.13A, where a total of 72 SF operations are required with a predicted cycle time of 2 minutes and 37 seconds. During the actual machine operation, the recipe file (Figure 3.13C), is uploaded to the PLC and requires 2 minute and 39 seconds to execute. Finally, as in case study 1, the cumulative time for generating the required recipe file is observed to be less than one second.

(A)

Shop Drawing	
No. of Operations	72
Frame Dimensions (mm)	Length (x) = 3,048 Height (y) = 2,439 Width (z) = 92
Collision Detection	N/A
Predicted Time (min:sec)	02:37
Actual Time (min:sec)	02:39

(B)



(C)

Recipe File

```
Main.frame_length_x,3048
Main.frame_height_y,2439
Main.frame_width_z,92
entrycount,72
VIS_File_Name,'pane12'
stringArray[0],'TL X:21 Y:25 Z:95'
stringArray[1],'TR X:21 Y:2413 Z:95'
stringArray[2],'BL X:21 Y:25 Z:0'
stringArray[3],'BR X:21 Y:2413 Z:0'
stringArray[4],'TL X:268 Y:25 Z:95'
stringArray[5],'TR X:268 Y:2413 Z:95'
stringArray[6],'BL X:268 Y:25 Z:0'
stringArray[7],'BR X:268 Y:2413 Z:0'
stringArray[8],'TL X:310 Y:25 Z:95'
stringArray[9],'TR X:310 Y:2413 Z:95
Y:2160 Z:95'
…
```

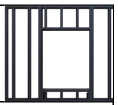
Figure 3.13 Case study 2 summary table (A); top-down tool-path output; and recipe file output excerpt (C)

3.4.3 Case Study 3

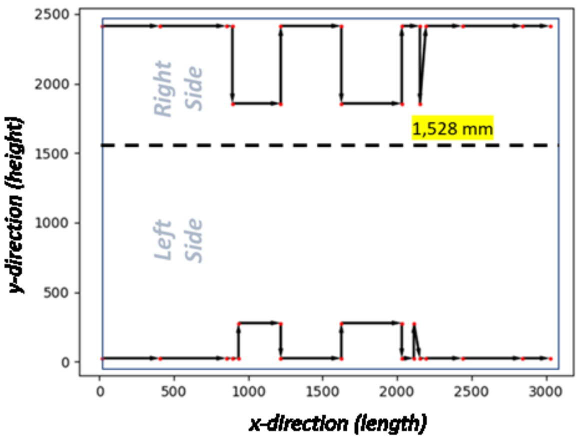
For the third case study, production of a wall panel with a single window opening is investigated. The stud spacing remains at 16 in, but changes are made to the overall frame dimensions to illustrate the variability in panel dimensions. Moreover, the addition of a window adds eight cripple studs distributed above and below the headers with two king studs and two jamb studs providing support for the opening (Figure 3.14A). These structural changes result in even distribution of SF operations among each of the four SFCs. Due to the large window opening, no SF operations are required within the buffer zones thereby allowing for a well-structured manufacturing strategy as illustrated in

Figure 3.14B. A brief description of the panel is presented in Figure 3.14A, where a total of 60 SF operations are required with a predicted cycle time of 1 minute and 44 seconds. During the actual machine operation, the recipe file (Figure 3.14C) is uploaded to the PLC and requires 1 minute and 48 seconds to execute. Finally, as in the previous two case studies, the cumulative time for generating the required recipe file is less than one second.

(A)

Shop Drawing	
No. of Operations	60
Frame Dimensions (mm)	Length (x) = 2,927 Height (y) = 2,607 Width (z) = 92
Collision Detection	N/A
Predicted Time (min:sec)	01:44
Actual Time (min:sec)	01:48

(B)



(C)

Recipe File

```
Main.frame_length_x,2927
Main.frame_height_y,2607
Main.frame_width_z,92
entrycount,60
VIS_File_Name,'panel3'
stringArray[0],'TL X:21 Y:19 Z:92
Y:57 Z:92'
stringArray[1],'TR X:21 Y:2588 Z:92
Y:2550 Z:92'
stringArray[2],'BL X:21 Y:19 Z:0 Y:57
Z:0'
stringArray[3],'BR X:21 Y:2588 Z:0
Y:2550 Z:0'
stringArray[4],'TL X:611 Y:38 Z:92'
stringArray[5],'TR X:611 Y:2569 Z:92'
stringArray[6],'BL X:611 Y:38 Z:0'
stringArray[7],'BR X:611 Y:2569 Z:0'
stringArray[8],'TL X:1180 Y:38 Z:92'
```

Figure 3.14 Case study 3 summary table (A); top-down tool-path output; and recipe file output excerpt (C)

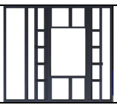
3.4.4 Case Study 4

For standard prefabricated home plans, the above three case studies represent most of the wall panel designs such that the above configurations can be combined in various forms without deviating from the essence of previous manufacturing strategies. However, in some instances, additional design changes may require the inclusion of lateral bracings for load bearing walls or small components positioned within the left and right buffer zones. These special conditions have a high probability of causing collision between SFCs while working within buffer zones.

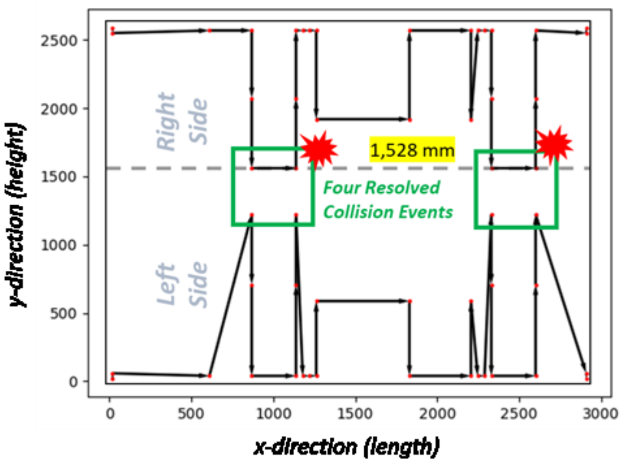
To validate the collision prevention approach presented earlier, a modified version of case study 3, involving eight lateral bracings, four of which are positioned on the left side and four on the right side of the existing window opening (Figure 3.15A), is investigated. The revised frame design is constructed by editing the exported CSV file from the previous case study. The modifications are made such that the carriages will collide if the sequence of operations remains as illustrated in case study 3. In short, the middle two bracings are placed 338 mm apart where the shortest allowed centerline distance between two carriages must exceed 340 mm. As predicted, four collision points relating to lateral bracings are identified, wherein three potential solutions are proposed with the best solution obtained from avoidance strategy 6 (Figure 3.15B). The reassessed tool-paths reveal that the right side remains unchanged while the left side is modified to avoid potential collisions. A brief description of the panel is presented in Figure 3.15A, where a total of 108 SF operations are required with a predicted cycle time of 3 minutes and 21 seconds.

Running the actual recipe file (Figure 3.15C) on the prototype machine produces a cycle time of 3 minutes and 26 seconds and confirms the proposed collision avoidance methodology. For instance, Figure 3.15D(i) depicts all four SFCs operating near the frame edges initially, whereas when the frame is positioned for fastening of lateral bracings, the left-side SFCs move to the left buffer zone while the right-side SFCs continue working within the right working zone as presented in Figure 3.15D(ii). As the sequence continues, all the carriages progress in a parallel fashion such that the left SFCs clear the buffer zone before the arrival of right SFCs in the right buffer zone as observed in Figure 3.15D(iii). Finally, the cumulative time for generating the required recipe file is approximately one second; however, the prior modifications to the exported CSV file required an additional 2 minutes.

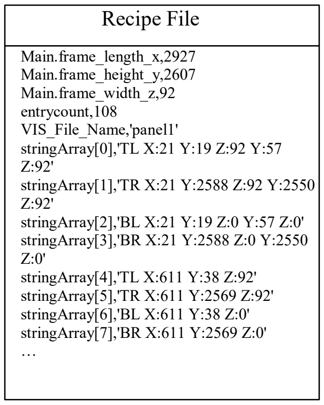
(A)

Shop Drawing	
No. of Operations	108
Frame Dimensions (mm)	Length (x) = 2,927 Height (y) = 2,607 Width (z) = 92
Collision Detection	Four potential collision events
Predicted Time (min:sec)	Strategy 4: 03:27 Strategy 5: 03:26 Strategy 6: 03:21
Actual Time (min:sec)	03:26

(B)



(C)



Recipe File

```
Main.frame_length_x,2927
Main.frame_height_y,2607
Main.frame_width_z,92
entrycount,108
VIS_File_Name,'panel1'
stringArray[0],'TL X:21 Y:19 Z:92 Y:57
Z:92'
stringArray[1],'TR X:21 Y:2588 Z:92 Y:2550
Z:92'
stringArray[2],'BL X:21 Y:19 Z:0 Y:57 Z:0'
stringArray[3],'BR X:21 Y:2588 Z:0 Y:2550
Z:0'
stringArray[4],'TL X:611 Y:38 Z:92'
stringArray[5],'TR X:611 Y:2569 Z:92'
stringArray[6],'BL X:611 Y:38 Z:0'
stringArray[7],'BR X:611 Y:2569 Z:0'
…
```

(D)

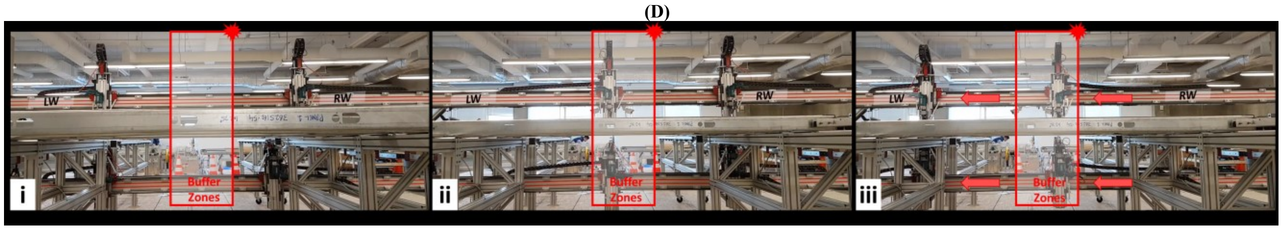


Figure 3.15 Case study 4 summary table (A); top-down tool-path output; recipe file output excerpt (C); and real-world operation (D)

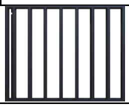
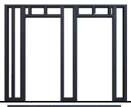
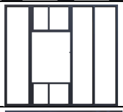
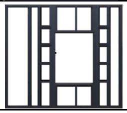
3.4 Discussion

From the four case studies outlined above, it can be observed that depending on the wall panel design, various tool-path configurations can be generated to meet the manufacturing requirements. Since most wall designs do not require multiple concurrent operations within both left and right buffer zones, the proposed methodology provides near optimal tool-paths. This fact can be seen in case studies 2 and 3, where no backtracking and no excessive tool traveling is observed. Nevertheless, even when concurrent operations occur, i.e., case study 4, the proposed methodology can modify the tool-paths to avoid potential collisions.

The proposed framework allows for fast generation of recipe files using manufacturing information outputted by information-rich 3D-BIM models. The system also provides cycle times that are comparable to the actual screwing process (provided in Table 3.2). Moreover, the applicability of in-shop modifications can be observed in case study 4, while case study 3 is modified to include eight lateral bracings. This change in wall design results in four potential collision points thus allowing for the successful initialization of collision avoidance strategies.

Since the cycle time for ball screws is assumed to be constant, as presented in Table 3.1, the time distribution between the functional and non-functional paths can be summarized. Here, the functional time can be considered as operational time dedicated to SF operations, whereas the non-functional time constitutes all the non-operating times, i.e., time required for positioning either the frame or carriages. Table 3.2 presents the effect of varying screw distribution on the non-operating (non-functional) time. For instance, functional time for right-side SF operations is always greater than left-side SF operations. This fact holds true even when the operations are split evenly between each SFC as seen in case studies 1 and 3. In short, the right-side SFCs on average require 10% additional functional time due to their higher ball screw cycle times as indicated in Table 3.1. However, for case study 2, the exceptionally high variance between the right and left non-operating time is the result of the highly uneven distribution of SF operations resulting from the addition of two doors.

Table 3.2 Summary of case studies, including a breakdown of screw-fastening operations and cycle time analysis

Case Study	Shop Drawing	Predicted Time (min:sec)	Breakdown of Screw-Fastening Operations					Actual Time (min:sec)	Operating and Non-operating Time analysis		
			TL	TR	BL	BR	Total			L	R
1		01:00	9	9	9	9	36	01:03	Operation Time (sec)	38	45
									Non-operating %	36	25
2		02:37	11	25	11	25	72	02:39	Operation Time (sec)	47	126
									Non-operating %	71	21
3		01:44	15	15	15	15	60	01:48	Operation Time (sec)	64	75
									Non-operating %	41	30
4		Str. 4: 03:27 Str. 5: 03:26 Str. 6: 03:21	27	27	27	27	108	03:25	Operation Time (sec)	114	136
									Non-operating %	44	34

3.5 Conclusions

The rise in offsite construction aims to address the inefficiencies of onsite construction by shifting construction to a controlled, indoor factory environment. Reinforced by several benefits, which include increased quality of the final product and project deliverability as well as reduced impact of labour shortages, poor weather conditions, and material waste, this shift to a factory environment demands an increased level of automation. To support such a shift, a prototype machine is being tested at the University of Alberta in Edmonton, Canada, which aims to assist in assembling and fastening of LGS framed wall-panels. Considering limited availability of similar research material, the presented research explores a framework for transferring the manufacturing information from 3D-BIM to an open-source file format readable by the PLC used to control the prototype machine. The proposed framework consists of three unique stages: (1) interpreter; (2) optimizer; and (3) post-processor, which work together to generate automatic collision-free tool trajectories. Here, generation of such trajectories requires in-depth integration of collision detection and avoidance solutions based on DES. Moreover, an open-source recipe format is proposed for the operation of the machine, which allows for greater expandability, modularity, and openness for future design modifications or for adopting such a format for other related machines. The time required for converting CAD information to recipe format is found to be approximately one second, which remains relatively constant even in cases of complex optimization resulting in excessive collision points.

The proposed framework is implemented as a Python-based program, and the results from various panel configurations, with and without building apertures, are presented for validating the accurate information transfer between the 3D-BIM output to real-world machine execution. In future, the proposed framework will be implemented with a vision-based system for initial quality control as well as dynamic collision prevention and path optimization.

Chapter 4 Tool-path generation for automatic manufacturing of light-frame panels: Wood[3]

4.1 Introduction

Similar to light-gauge steel framing, wood framing is also extensively used in construction practice today. In fact, the North American residential construction industry utilizes wood as its primary construction material. Nevertheless, similar problems exist to those experienced by the light-gauge steel framing industry, wherein conventional construction has become stagnant and innovative technologies are required to overcome the current technical limitations (Li 2016).

This chapter expands on the research presented by Li (2016), who proposed a process for flexible residential wall panel manufacturing. This process included a detailed overview of a prototype modular machine to support automation of the multi-panel manufacturing. The proposed machine is design to perform three unique operations—i.e., nailing, drilling, and cutting—for multi-panel assembly. The "multi-wall panel" process constitutes assembling of multiple wall panels together as one single wall panel and separating these panels into individual wall panels at later stages of the panel prefabrication process. This process minimizes the machine set-up time, reduces material waste, and maximizes the utilization of the proposed machine. Additionally, fabrication in the form of multi-panels as an alternative to single walls also increases the utilization of downstream stations due to reduced idle time.

[3] The manuscript appearing as Chapter 4 of this thesis will be adapted for submission to an academic journal upon the completion of this thesis.

More specifically, the assembly of each multi-wall panel is conducted through the addition of a single stud at a time, with each stud automatically positioned and secured. Feeding studs one at a time allows for multi-walls to be extruded to lengths in excess of 20 ft without requiring an exceptionally large footprint for the frame assembly process. Figure 4.1 shows an overview of the wood framing machine, which is composed of two sets of dragging jaws and four modular stations—feeding, nailing, cutting, and drilling—on each side.

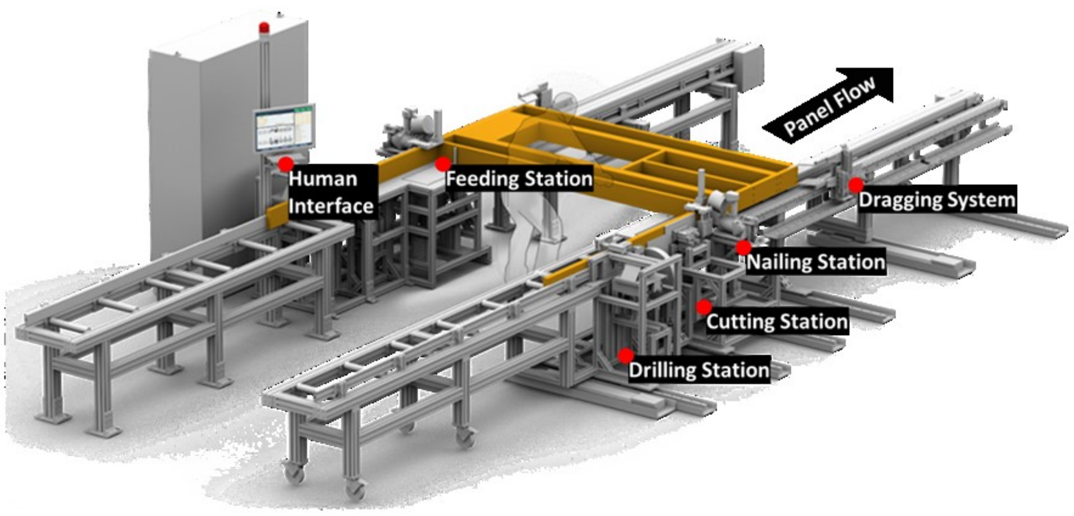


Figure 4.1 Wood Framing Machine Prototype Overview

The dragging jaws are responsible for moving the panel to the location of the next operation, which can be the addition either of a stud or of a completed wall component. The movement of the frame allows the worker to remain stationary at the feeding station and feed the components as required. The frame processing operations, i.e., nailing, cutting, and drilling, are conducted as per the recipe file, with the importance of each operation explained below:

4.1.1 Nailing Station

The nailing station consists of a pneumatic nailing gun with two degrees of freedom consisting of up-and-down and back-and-forth motions. The up-and-down motion is used to position the nailing gun at the appropriate height for nailing wall studs and components to the top- and bottom-plates, whereas the back-and-forth motion allows for triggering of actual nailing operation. The nailing station can accommodate either normal or vertical studs depending on the frame confirmation (denoted by the red rectangle in Figure 4.2).

4.1.2 Cutting Station

The cutting station consists of a circular saw with a single degree of freedom in the vertical (up-and-down) direction. Since the proposed wood framing machine is optimized for producing multi-wall panels, the cutting station partially precuts the plates such that each multi-wall panel can be easily separated into single panels at the on-site assembly stage (denoted by the black line in Figure 4.2). Moreover, the cutting station also cuts the extra top/bottom plate which remains after the completion of a given multi-panel.

4.1.3 Drilling

As with the nailing station, the drilling station consists of a drill with two degrees of freedom consisting of up-and-down and back-and-forth motions. The up-and-down motion is used to position the drill at the appropriate plate height, whereas the back-and-forth motion allows for through hole drilling. The addition of multiple holes on the top/bottom plates eases the process of lifting the panels at downstream stations in the factory environment, and later the process of transferring completed components to the site.

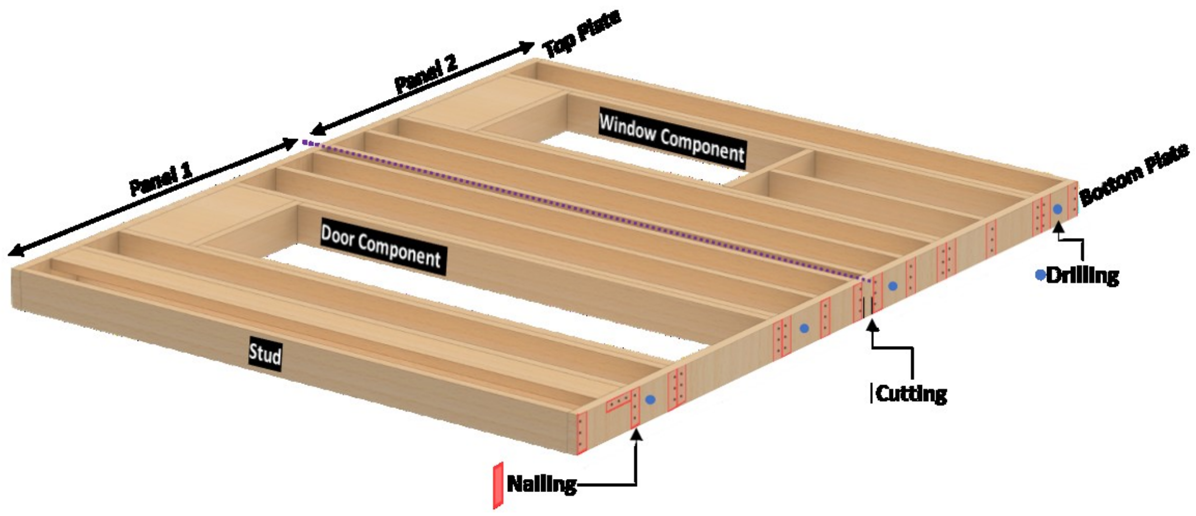


Figure 4.2 Sample multi-wood panel configuration

Since each of the above stations is in a fixed position, the frame must be moved to each location where a given operation is to be conducted. To facilitate the movement of the dragging jaw, motion planning for each operation is critical. The following section adopts the MMRW framework from LGS framing in order to generate a recipe file which can be read by the PLC to execute all the aforementioned operations.

4.2 Wood wall-framing machine motion planning

Similar to light-gauge steel framing, the proposed MMRW framework can be adopted for generating tool paths for a wood-framing machine. Each of the three stages in Section 3.3 are explored again here, with changes introduced only within each of the main modules. Here, the changes are generally resulting from the multi-spindle configuration wherein each operation (i.e., nailing, cutting, and drilling) requires a separate operational axis with varying offsets (Figure 4.3). Furthermore, since the stations are positioned on the frame edges, no operations are required beyond the top and bottom plates.

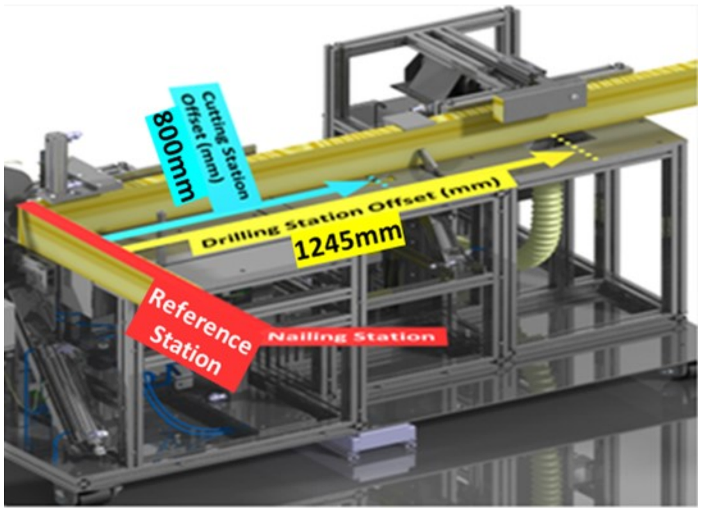


Figure 4.3 Wood Framing station with respective offsets

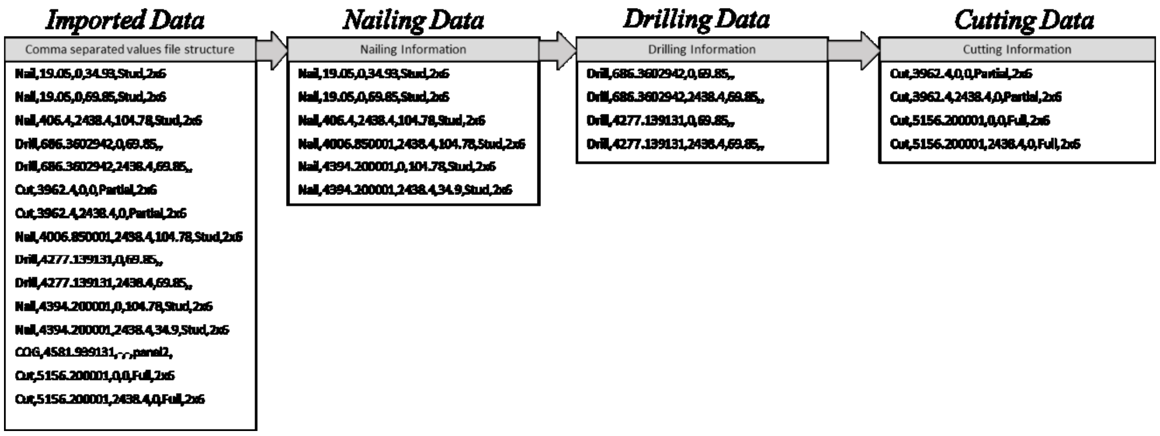
As with steel framing, here the proposed framework is programmed using Python (Version 3.6.4) because of its ability to quickly modify numerical data, as well as the relative ease with which it reads, processes, and manipulates external information. Furthermore, to implement machine constraints, the MMRW framework is built upon Cartesian movements, which include bidirectional movement of nailing guns, drills, and table saws in the ±*z*-direction, and unidirectional frame transfer in the +*x*-direction. The following subsections highlight the main modules contained within each of the three stages as shown in Figure 3.2.

4.2.1 Data Input

A data input module allows for importing and reading of a CSV file through file browser and data acquisition functions. The file browser function provides a user interface for selecting a file path and allows for importing of raw data files. The data acquisition function gathers the information relating each operation within one nested array, and the panel's overall dimensions and panel ID are stored in separate array and variable, respectively, for the post-processor stage.

4.2.2 Data Extraction

The data extraction module follows the same processes as described in section 3.3.2, wherein the nested array from section 4.2.1 is separated into three sub-arrays containing filtered information for nailing, cutting, and drilling (Figure 4.4). As shown in Figure 4.3, separate operational axes with varying offsets also result in "dead-zones", where certain operations are considered infeasible due to physical limitations imposed by the multi-spindle configuration. For instance, all cutting operations located within first 800 mm or drilling operations within 1,245 mm are impractical due to the unidirectional movement of any given wall frame. To avoid the inclusion of "dead-zone" data, the data extraction module also removes any data point associated with drilling or cutting in dead-zones. The data stored in each array is assumed to be in the correct units as presented in the original CSV file.



```
Nail,19.05,0,34.93,Stud,2x6
Nail,19.05,0,69.85,Stud,2x6
Nail,406.4,2438.4,104.78,Stud,2x6
Drill,686.3602942,0,69.85,,
Drill,686.3602942,2438.4,69.85,,
Cut,3962.4,0,0,Partial,2x6
Cut,3962.4,2438.4,0,Partial,2x6
Nail,4006.850001,2438.4,104.78,Stud,2x6
Drill,4277.139131,0,69.85,,
Drill,4277.139131,2438.4,69.85,,
Nail,4394.200001,0,104.78,Stud,2x6
Nail,4394.200001,2438.4,34.9,Stud,2x6
COG,4581.999131,-,-,panel2,
Cut,5156.200001,0,0,Full,2x6
Cut,5156.200001,2438.4,0,Full,2x6
```

```
Nail,19.05,0,34.93,Stud,2x6
Nail,19.05,0,69.85,Stud,2x6
Nail,406.4,2438.4,104.78,Stud,2x6
Nail,4006.850001,2438.4,104.78,Stud,2x6
Nail,4394.200001,0,104.78,Stud,2x6
Nail,4394.200001,2438.4,34.9,Stud,2x6
```

```
Drill,686.3602942,0,69.85,,
Drill,686.3602942,2438.4,69.85,,
Drill,4277.139131,0,69.85,,
Drill,4277.139131,2438.4,69.85,,
```

```
Cut,3962.4,0,0,Partial,2x6
Cut,3962.4,2438.4,0,Partial,2x6
Cut,5156.200001,0,0,Full,2x6
Cut,5156.200001,2438.4,0,Full,2x6
```

Figure 4.4 Data sorting for wood framing operations

4.2.3 Path Generation (Multi-spindle Tool-path Generation Algorithm)

The path generation module produces initial tool-paths for each operational category (i.e., nailing, sawing, drilling) independently. The path generation of nailing starts with sorting of the *x*-value, followed by sorting of the *z*-values, both in ascending order, with the resulting array labeled "*sortedNData"*. The sawing and drilling paths are sorted in ascending fashion as well, but, since there can only be one *z*-value per given *x*-coordinate, no additional sorting is required. The sorted data for sawing is sorted in a "*sortedCUData*" array, whereas the drilling information is sorted in a "*sortedDRData*" array.

Since the wood-framing machine has a unidirectional flow of wall panels, each of the required operations must be executed at the precise location without backtracking. The multi-spindle configuration thus requires the order of operations to be arranged to allow for smooth unidirectional execution. To account for varying offsets between stations, the nailing station is chosen as a reference station, which in turn is used to sequence each operation in order of intended manufacturing information as given in the frame file. Since each operation is already pre-planned, a custom path integration algorithm is designed to rearrange each operational path into one encompassing path for the final operation of the wood framing machine. The path integration algorithm functions as follows:

1. *Set Drilling Offset (DO) = 1,245 mm and Cutting Offset (CO) = 800 mm*
2. *Set i, k, and l as 0 representing the first index of arrays sortedNData, sortedDRData, and sortedCUData, respectively*
3. *Create an empty "sortedDataList" array*
4. *Find the operation with minimum* x-*value:*
    - *Min(sortedNData[i][1], sortedDRData[k][1] + DO, sortedCUData[l][1] + CO)*
5. *Increment the index with corresponding operation:*

- *If the minimum* x-*value is index 0, then*
    - *$i = i + 1$*
- *If the minimum* x-*value is index 1, then*

    $k = k + 1$
- *If the minimum* x-*value is index 2, then*

    $l = l + 1$

6. *Append the operational details to "sortedDataList", including all the Cartesian coordinates, operation type, and other corresponding information*
7. *Repeat from step 1, until all operations are sorted and appended to "sortedDataList"*

The conclusion of the path generation module will output a comprehensive array with each line representing a unique operation and its respective order of execution (See Figure 4.5).

```
['Nail', 19.05, 0.0, 34.93, 'Stud', '2x6']
['Nail', 19.05, 2438.4, 34.93, 'Stud', '2x6']
['Nail', 19.05, 0.0, 69.85, 'Stud', '2x6']
['Drill', 3124.760294, 0.0, 69.85, '', '']
['Drill', 3124.760294, 2438.4, 69.85, '', '']
...
['Cut', 3962.4, 0.0, 0.0, 'Partial', '2x6']
['Cut', 3962.4, 2438.4, 0.0, 'Partial', '2x6']
['Nail', 3251.2, 0.0, 34.93, 'Stud', '2x6']
['Nail', 3251.2, 2438.4, 34.93, 'Stud', '2x6']
...
['Nail', 4006.850001, 0.0, 104.78, 'Stud', '2x6']
['Nail', 4006.850001, 2438.4, 104.78, 'Stud', '2x6']
['Cut', 5156.200001, 0.0, 0.0, 'Full', '2x6']
['Cut', 5156.200001, 2438.4, 0.0, 'Full', '2x6']
['Nail', 4394.200001, 0.0, 34.93, 'Stud', '2x6']
...
```

Figure 4.5 Sample Format of "sortedDataList"

4.2.4 Simulation: Integration of schedule (time) information

The path simulation module is designed to simulate tool-paths using discrete-event simulation (DES), where the basic machine logic mimics a real-world system. However, for the wood framing machine the time-based information is not critical since each wood stud is fed separately and requires a worker to be present. The manual nature of this operation, combined with the fact that each spindle is physically mounted on a fixed platform, negates the need for predicting the time. In other words, no tool can collide with any other tool since each tool is fixed on either the right or left side, with offsets in place between each of the tools on a given side.

4.2.5 Collision Detection and Avoidance

As explained above, the wood framing machine will not encounter any collision between moving parts due to its unique multi-spindle design, such that collision detection and avoidance functionalities are not required.

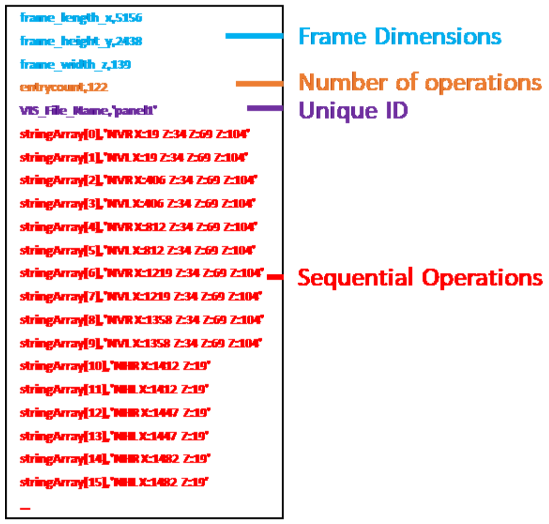
4.2.6 Post-processing

The post-processing module is designed to output a recipe file that can be interpreted by the PLC as motion commands to operate the wood framing machine. As with the steel framing, this module also generates a CSV file that outputs manufacturing information in a user-friendly manner, allowing for in-shop modifications. Here, the user-modified file can be fed back into the interpreter for the generation of modified recipe files.

The module for generating the final recipe file consists of four primary functions comprising data acquisition, path sorting, data formatting, and output file formatting. The data acquisition function gathers the required information relating to frame dimensions, number of operations, frame ID, and sorted tool-path. This information is transferred to the path-sorting function, which formats the data into ten unique categories as shown in Table 4.1. The sorting function also ensures that the final data excludes all Cartesian information relating to "*y*-values", which, in-turn, is represented by either "R" or "L" appearing as the third letter in the code for each of the ten unique categories, representing y = 0 and y > 0, respectively. The path-sorting function organizes data such that all the information relating to a given operation at a particular *x*-position is outputted in a single line starting with operational ID, followed by *x*-position and by one or more *z*-positions (see Figure 4.6, "Sequential Operations"). This formatting method allows the PLC to read the recipe file and quickly extract the information for each tool based on its operational ID. The path-sorting information is passed to the data formatting function, which formats the information as shown in Figure 4.6. Finally, the sorted information is outputted as a .rcp file, which is uploaded to the PLC for final execution.

Table 4.1 Wood Framing Machine—Operational Categories

CODE	OPERATIONAL CATEGORY	CODE	OPERATIONAL CATEGORY
NVR	Nailing Vertical – Right Side	NVL	Nailing Vertical – Left Side
NHR	Nailing Horizontal – Right Side	NHL	Nailing Horizontal – Left Side
DRR	Drilling – Right Side	DRL	Drilling – Left Side
PCR	Partial Cut – Right Side	PCL	Partial Cut – Left Side
FCR	Full Cut – Right Side	FCL	Full Cut – Left Side

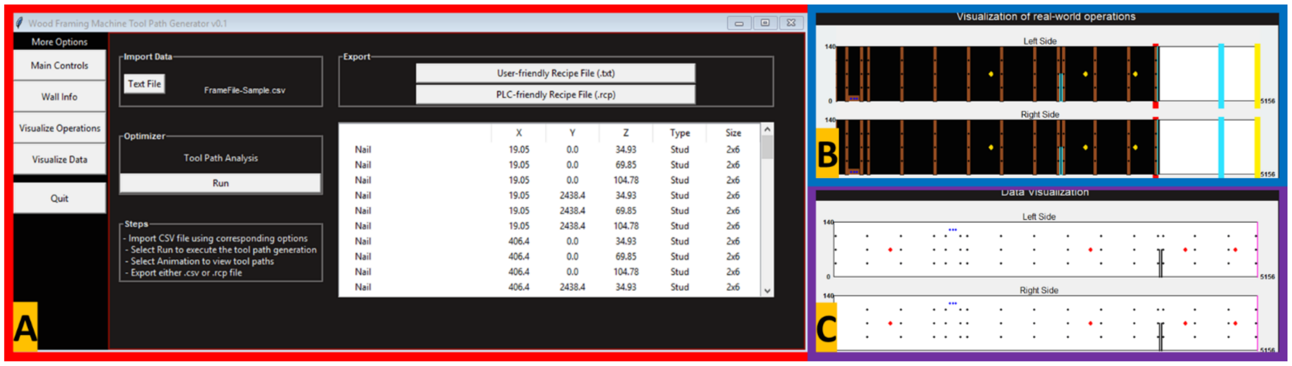


```
frame_length_x,5156
frame_height_y,2438
frame_width_z,139
entrycount,122
VIS_File_Name,"panel1"
stringArray[0],"NVR X:19 Z:34 Z:69 Z:104"
stringArray[1],"NVL X:19 Z:34 Z:69 Z:104"
stringArray[2],"NVR X:406 Z:34 Z:69 Z:104"
stringArray[3],"NVL X:406 Z:34 Z:69 Z:104"
stringArray[4],"NVR X:812 Z:34 Z:69 Z:104"
stringArray[5],"NVL X:812 Z:34 Z:69 Z:104"
stringArray[6],"NVR X:1219 Z:34 Z:69 Z:104"
stringArray[7],"NVL X:1219 Z:34 Z:69 Z:104"
stringArray[8],"NVR X:1358 Z:34 Z:69 Z:104"
stringArray[9],"NVL X:1358 Z:34 Z:69 Z:104"
stringArray[10],"NHR X:1412 Z:19"
stringArray[11],"NHL X:1412 Z:19"
stringArray[12],"NHR X:1447 Z:19"
stringArray[13],"NHL X:1447 Z:19"
stringArray[14],"NHR X:1482 Z:19"
stringArray[15],"NHL X:1482 Z:19"
_
```

Figure 4.6 Wood Framing Machine’s file structure of recipe file

4.2.7 GUI Design

The graphical user interface (GUI) for the MMRW framework consists of four tabs—main controls, wall info, visualize operations, and visualize data—as shown in Figure 4.7. The “main controls” tab includes options for uploading the CSV file, followed by execution of the optimizer that generates a static table with proposed tool-path (Figure 4.7, A). Subsequently, export options can output recipe files for execution on the WFMP, as well as CSV files for in-shop modifications. The “wall info” tab shows the frame dimensions and panel ID as indicated in Section 4.3.1. Finally, the “visualize operations” tab animates the sequential operations of the actual machine (Figure 4.7, B). This tab simulates the recipe file in a virtual environment. Finally, the “visualize data” tab shows an animation of all the required operations as inputted by the user (Figure 4.7, C).



Steps
- Import CSV file using corresponding options
- Select Run to execute the tool path generation
- Select Animation to view tool paths
- Export either .csv or .rcp file

	X	Y	Z	Type	Size
Nail	19.05	0.0	34.93	Stud	2x6
Nail	19.05	0.0	69.85	Stud	2x6
Nail	19.05	0.0	104.78	Stud	2x6
Nail	19.05	2438.4	34.93	Stud	2x6
Nail	19.05	2438.4	69.85	Stud	2x6
Nail	19.05	2438.4	104.78	Stud	2x6
Nail	406.4	0.0	34.93	Stud	2x6
Nail	406.4	0.0	69.85	Stud	2x6
Nail	406.4	0.0	104.78	Stud	2x6
Nail	406.4	2438.4	34.93	Stud	2x6

Figure 4.7 Sample GUI for wood framing machines

4.3 Conclusion

As with the SFMP, here multiple frame files are tested using the proposed framework, and each resulting recipe file is successfully executed by the WFMP. The proposed MMRW framework is shown to be versatile for both steel and wood framing. The real-world implementation using a Python-based program is also proven to be flexible for showcasing information in a user-friendly manner.

Chapter 5 Improved automated wall framing[4]

5.1 Introduction

Cold-formed steel (CFS) frames are commonly used for constructing interior and exterior wall frames for mid-size residential and commercial projects, and in fact in many jurisdictions building codes require the use of CFS in structures taller than 6-storeys. However, this method of construction usually requires excessive manual work including handling of heavy steel pieces. In addition to safety and ergonomic issues, these manual operations also result in lower productivity, reduced quality of work, and increased material waste. These challenges associated with traditional construction can be circumvented by adopting modern methods of construction that promote tangible benefits through pre-fabricated construction (Lawson & Ogden 2008).

Prefabricated construction provides a cost-effective solution, with most of the building components produced off-site in a controlled factory environment. Paudel et al. (2016) show that the use of prefabrication and preassembly has increased by 86% over the past 15 years. They also pointed out that prefabricated construction provides significant cost and time savings. A case study (Shahzad et al. 2015) compares 66 building projects and finds that prefabrication on an average leads to a 34% reduction in completion time and a 19% reduction in completion costs. Through increased automation and innovation, such advantages can further the adoption of modern methods of construction (Bock 2015).

[4] The manuscript appearing as Chapter 5 of this thesis was published in *Proceedings, 2018 Modular and Offsite Construction Summit*, Hollywood, FL, USA, Mar. 22–25.

Increased demand for higher product variability and shorter cycle times within the manufacturing industry has paved the way for extensive research around dynamic decision-making support systems for existing products (Ahmad & Plapper 2015; Ahmad et al. 2016), and increased application of Design for Excellence (DFX) methods for product design. Design for manufacturing (DFM) and Design for production (DFP) are two well-studied methodologies of DFX that involve cycle time analysis, where DFM aims to ensure the manufacturability of a component as per supplier's capability, while DFP involves evaluating manufacturing capacity and manufacturing time (Herrmann & Chincholkar 2000). As shown by Herrmann (2003) DFP and DFM techniques can improve manufacturing systems such as production lines, factories, and supply chains, where these techniques function as decision-making tools for further system enhancements. However, application of such systems in the construction industry has been lacking.

Nevertheless, advances in off-site construction are promoting greater use of simulation modelling. In particular, Discrete-event simulation (DES) is a computer-based simulation approach wherein real-world systems are converted to discrete events mimicking real-world processes. Many studies have been developed that apply DES to improve the production line for modular construction (Altaf et al. 2014, 2015; Liu et al. 2015). Simphony has been used extensively to construct such simulation models due to its close association with the construction domain, where it provides tools for assessment of project duration, resource utilization, and general decision making support (Afifi et al. 2016).

To date, the use of virtual simulation as a decision-making support tool during the prototype phase of automated construction machinery has been limited. Therefore, this research aims to combine DES with the manufacturing reduction framework, MTTP, proposed by Johnson (2003) in order to understand the manufacturing capacity and manufacturing time of a steel framing machine prototype (SFMP). In short, this research proposes a cycle time study in which various modifications to an existing light-gauge steel (LGS) wall panel fabrication machine prototype are investigated. Initially, the real-world machine logic is mapped using DES to validate the accuracy of utilizing such modelling techniques; moreover, the model is further used as a baseline to forecast the effect of proposed design changes. Each design modification aims to increase productivity through the removal of existing bottlenecks, where the results from various panel configurations can be used as a guideline informing future in-depth design modifications.

5.2 System Description

Figure 5.1 shows a brief overview of the main components of the steel framing machine prototype (SFMP). This machine follows a multi-phase semi-automated framing approach for constructing LGS wall-framed panels, wherein panel assembly is undertaken manually, and the panel fastening is automated. More specifically, the process starts with uploading of manufacturing plans via a human-machine interface (HMI), which allows the machine to adjust its dimensions to allow for the panel assembly. The process of manual assembly is undertaken on Table A, with the assembly time approximated using the panel assembly Equation 5.2.1 from Liu et al. (2015):

$$T_A = \sum_{i=1}^{N_{ST}} T_{ST} + T_{BP}N_{BP} + T_{TP}N_{TP} + T_w N_w + T_D T_D + V \qquad (5.2.1)$$

where TA = total assembly time; TBP = time to place bottom plate; TTP = time to place top plate; TST = time to place a single stud; TW = time to assemble window; TD = time to assemble door; V = variation in assembly process; and NBP, NTP, NST, NW, and ND = number of subcomponents relating to the aforementioned variables.

The completion of the manual assembly phase is followed by an automatic screw-fastening phase, where the soft-connected panel is squared and dragged via electromagnetic squares arranged in a rectangular pattern along the corners of the assembled panel. Synchronized dragging allows the frame to be precisely positioned between the two horizontal beams of a stationary gantry, wherein four screw-fastening carriages operate in unison to place screws in predetermined locations. Considering that the carriages work in pairs, the movement of carriages is preplanned by a custom tool path algorithm designed to avoid collisions and to allow for rapid screwing operations. Since the carriages are located on a stationary gantry, the frame is repositioned to each location requiring screw-fastening operations. Finally, the automated phase ends when the framed wall panel arrives on Table B and is offloaded for further processing.

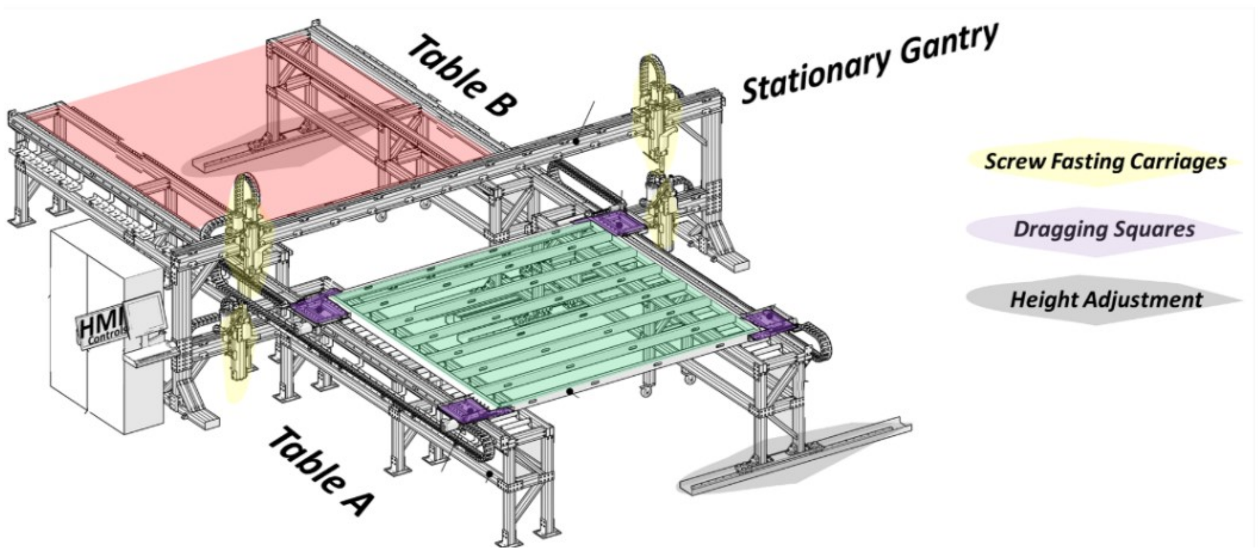


Figure 5.1 Overview of Steel Framing Machine.

5.3 Model Construction

Since the panel configuration varies considerably from one panel to another, the simulation model is linked to an MS Access database, which provides an accurate link between the model and the actual shop drawings. The following sections describe the process of utilizing the shop drawing information to construct a reliable simulation model for the production of each steel panel.

Simulation models are constructed in the general template of Simphony using DES. In Simphony, each main operation is modelled using a “Task” element, which holds an entity for a specified amount of time. The data flow between different tasks is controlled using “Activator” and “Valve” elements. For any “Task” elements requiring resources, a “Capture” element is utilized to occupy available resources for the specified duration. In addition to different elements, Simphony also features local and global variables for representing attributes of entities at individual level and at simulation level, respectively. A select list of variables and their descriptions is presented in Table 5.1 as an example.

Table 5.1 Summary of select variables from simulation models (Local = L and Global = G).

Name	Description	Name	Description
LX(1)	*x*-position	GX(97)	Right side screw operations at given *x*
LX(2)	*y*-position	GX(98)	Left side screw operations at given *x*
LX(3)	*z*-position	GX(99)	set *x*-position after frame positioning
LN(97)	Panel ID	GX(12)	Panel ID Tracker
LX(4/)	No. of Windows	LX(5/6)	No. of Doors/ Studs

The simulation sequence begins with the creation of all the entities stored in the database. Initially, the value for GX(12) is set to 1, thereby allowing the entities corresponding to that panel ID to flow into the soft connection phase wherein a "worker" resource is acquired for the duration calculated using Eq. 1. Following this phase, the soft-connected panel is moved as per LX(1) and the said position is stored in GX(99). Here, screw-fastening operations relating to GX(99) are simulated using closed loops as represented by blue lines in Figure 5.2. The completion of screw-fastening operations at a given frame position is simulated by passing two entities through an activator, which allows for the entities relating to the next frame position to pass (represented as red lines in Figure 5.2). This information is used to reposition the frame, after which the above process repeats until the completion of required operations. At the end of the screw-fastening phase, a short offloading delay is simulated, followed by the incrementing of GX(12) and activation of the original "valve", thereby allowing for information pertaining to the next frame to enter the simulation model (represented as green lines in Figure 5.2). A sample overview of the DES model is shown in Figure 5.3.

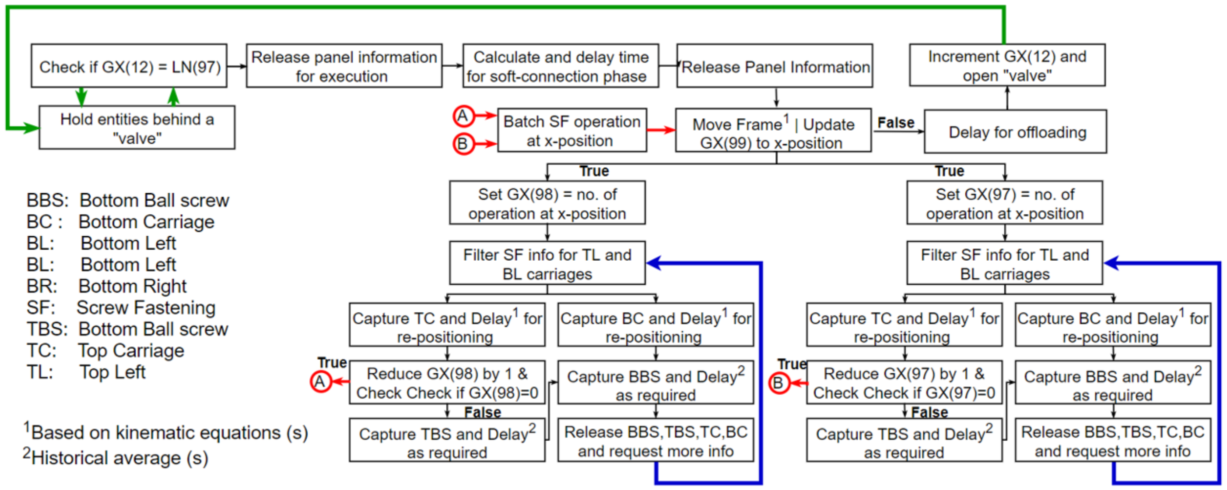

Check if GX(12) = LN(97)
Release panel information for execution
Calculate and delay time for soft-connection phase
Release Panel Information
Increment GX(12) and open "valve"
Hold entities behind a "valve"
Batch SF operation at x-position
Move Frame¹ | Update GX(99) to x-position
False
Delay for offloading
True
Set GX(98) = no. of operation at x-position
Set GX(97) = no. of operation at x-position
Filter SF info for TL and BL carriages
Capture TC and Delay¹ for re-positioning
Capture BC and Delay¹ for re-positioning
Reduce GX(98) by 1 & Check Check if GX(98)=0
Reduce GX(97) by 1 & Check Check if GX(97)=0
Capture BBS and Delay² as required
Capture TBS and Delay² as required
Release BBS,TBS,TC,BC and request more info
BBS: Bottom Ball screw
BC : Bottom Carriage
BL: Bottom Left
BL: Bottom Left
BR: Bottom Right
SF: Screw Fastening
TBS: Bottom Ball screw
TC: Top Carriage
TL: Top Left
¹Based on kinematic equations (s)
²Historical average (s)

Figure 5.2 Main Logic for Discrete-event Simulation Model.

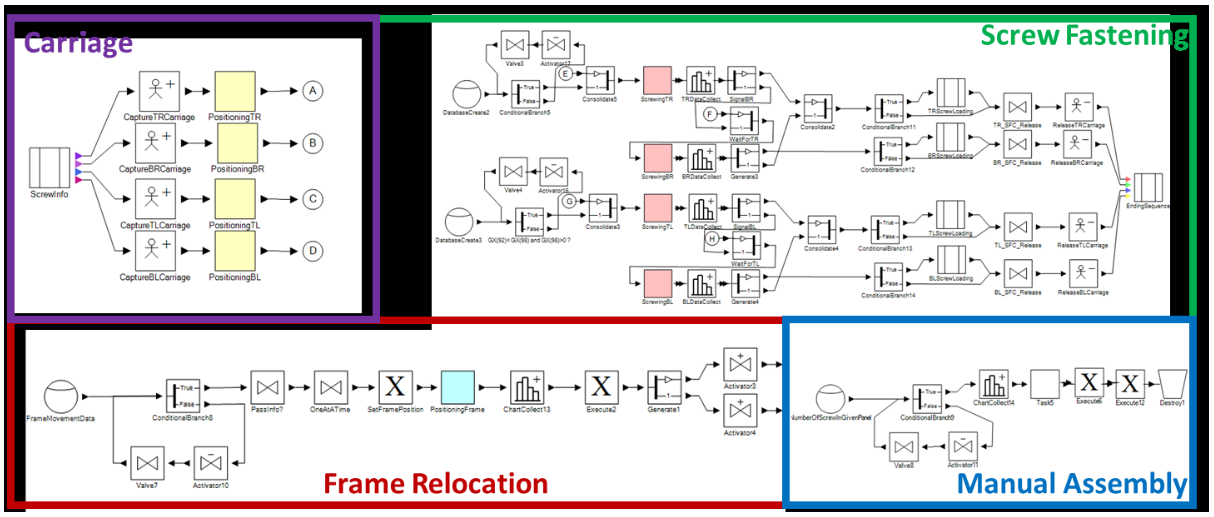


Figure 5.3 Simulation model of steel framing machine in Simphony.NET

5.3.1 Scenario 1: Manual Screw Feeding

For scenario one, each screwdriver can hold only 50 screws; once the magazine of screws has been exhausted, all operations are halted for the duration of the refilling phase. As per the current prototype design, the average duration for refilling a new magazine is assumed to be 50 seconds.

5.3.2 Scenario 2: Auto Screw Feeding System

For scenario two, the overall simulation sequence and machine logic remain the same as for scenario one. However, the screwdrivers are not limited to 50-screw magazines. This change requires the addition of a centralized screw feeding system that allows for automatic feeding of screws through feed tubes attached to a vibrating bowl. Here, vibrating bowls have a capacity of a few hundred screws and can be refilled easily. Moreover, the length of the feed tubes directly correlates to feeding rate, where an increase in tube length corresponds to a decrease in the feed rate. By estimating the relationship between cycle times and screw feeding rate, one can design mechanical systems that provide the most optimal cycle times and minimize costly design changes. This scenario closely resembles actions of “reduce waiting time per part” as proposed by Johnson (2003).

5.3.3 Scenario 3: Design Modification for Frame Clamping

For scenario three, another limitation resulting from the coupling of the top and bottom screwdrivers is investigated. Here, if the panel is not clamped from the top, the panel may be lifted by the force produced through the bottom screw-fastening operations. This high degree of interdependence, i.e., high coupling, thus requires the top screwdrivers to remain fully extended while the bottom operations are conducted, and negatively affects the cycle time during the hard connection phase. However, the addition of pneumatic clamping mechanisms can allow both screwdrivers to operate synchronously, thereby allowing for faster cycle times. This scenario closely resembles actions of “reduce processing time per part” as proposed by Johnson (2003).

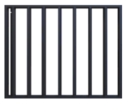
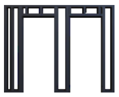
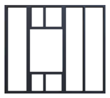
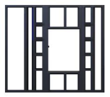
5.3.4 Scenario 4: Hybrid Design with Clamping and feeding

Since the design changes as stated in scenario two and three would require the addition of a pneumatic system, scenario four examines the potential of implementing both modifications working in tandem.

5.4 Model Input Data

To illustrate the effects of various panel configurations on cycle times, Table 5.2 shows a summary of selected panels with and without building apertures. Since each of the above scenarios necessitates changing a certain aspect of the machine's logic, the variability in panel designs is crucial for evaluating the effect on cycle time.

Table 5.2 Summary panels utilized for modelling cycle times.

Panel Design		1	2	3	4
Shop Drawing					
Frame Dimensions (mm)		Length (x) = 3048	Length (x) = 3048	Length (x) = 2927	Length (x) = 2927
* Width (z) = 92		Height (y) = 2439	Height (y) = 2439	Height (y) = 2607	Height (y) = 2607
Screw-fastening Operations	Left	9	11	15	27
	Right	9	25	15	27
	Total	36	72	60	108

5.5 Results and Discussion

Initially, the simulation model is run with the assumption of zero time associated with the assembly and offloading phase as well as infinite screw capacity. These assumptions allow for the basic validation of the simulation model as per real-world machine logic. In other words, the four panels as described in Table 5.2 are executed on the real-world machine and the resulting cycle times are compared to simulated times as shown in

Figure 5.4. Here, the simulated results are comparable to real-world results, thereby allowing for a baseline validation of the given scenarios.

Figure 5.5 shows the impact of four scenarios while simulating continuous production of 20 wall panels. Such production consists of simulating four panels from Table 2 five times with their respective soft-connection and offloading phases. Moreover, each simulation run is repeated 100 times with an identical seed value. Here, the repetition of simulation runs captures the probabilistic nature of soft-connection and offloading phases, whereas equal seed value allows for one-to-one comparison between the hard-connection phases of the scenarios. The total production time is found to vary from 3.1 to 2.4 hr. Since scenarios one and three require manual loading of screws, these scenarios are not affected by the feeding delays, whereas scenarios two and four reflect the negative impact of lower screw-feeding rates on the final production times. Without the inclusion of cost analysis, a comparison between each scenario is not viable. Nevertheless, by utilizing the simulation results, we can conclude that the addition of auto-feeding is only acceptable if the feeding delays are less than 6.3 seconds, whereas applying the top clamping and auto-feeding should only be adopted if the feeding delays are less than 8.8 seconds. By developing systems within the given constraints, the cycle times for SFMP can be either maintained or decreased, as shown in Figure 5.5. Furthermore, the addition of frame clamping with manual feeding (scenario three) can result in approximately a 13 percent drop in production times as compared to scenario one.

Finally, the reason for high production time for scenario one is shown in Figure 5.6, where the production of 20 panels requires 13 separate stops for the refilling phase. These excessive delays for manual loading thus result in scenario one being deemed the least desirable. Figure 5.7 shows an in-depth comparison among the various feeding rates and panel configurations, where the panels with closely positioned operations experience more significant effects of screw feeding delays.

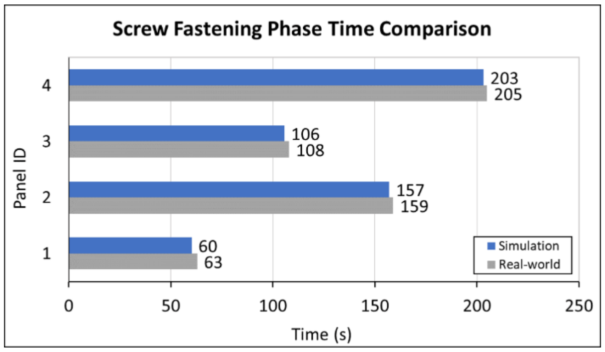


Figure 5.4 Screw-fastening time comparison between simulation model and real-world prototype.

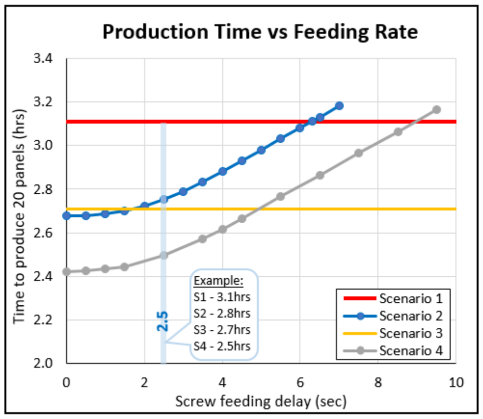


Figure 5.5 Comparison between time for production of 20 continuous panels and proposed scenarios.

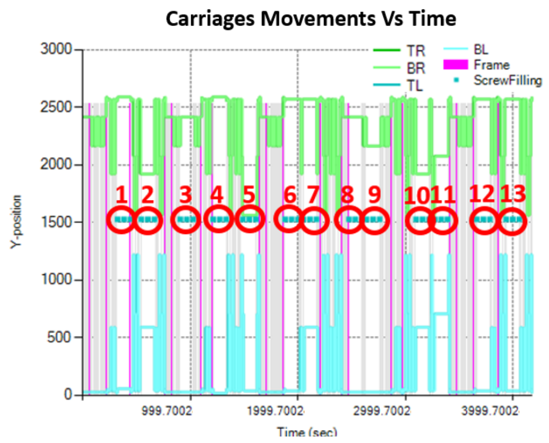


Figure 5.6 Carriage movements versus time comparison to produce 20 wall panels using manual screw loading (i.e., scenario 1).

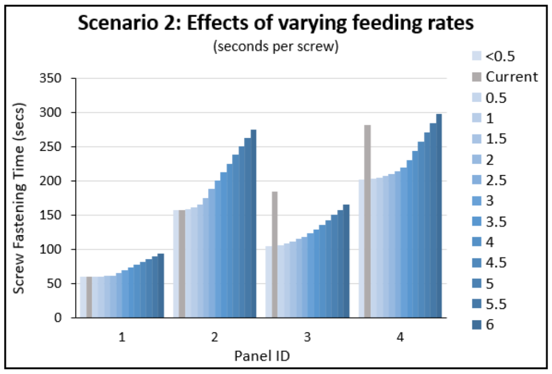


Figure 5.7 Understanding effects on cycle time for varying feed rates on scenario 2.

5.6 Conclusion

To improve cycle times during automatic screw-fastening operations, DES techniques are employed for estimating the implication of various system modifications. Such improvement studies have typically been limited primarily to the manufacturing industry; therefore, the contribution of this research is to apply throughput time reduction techniques for understanding the manufacturing capacity and manufacturing time for automated construction machinery. Through a preliminary model comparison with a real-world prototype machine, two bottlenecks relating to the coupling of screw-fastening operations and manual screw feeding are determined, while three modifications are investigated to minimize the impact of these bottlenecks. The simulation results provide valuable insights into potential modifications and their approximate effects on cycle time. For instance, for the screw-fastening phase, cycle time reductions of 13 percent or greater are possible by applying proposed modifications. As a further example, by changing the screw-feeding system from manual to auto-feeding, the benefits can be immense if the screw feeding rate is kept below 6.3 seconds, whereas exceeding this threshold would result in higher cycle times. Detailed cost and engineering information is required to better understand the opportunity cost of each proposed solution; however, compared to manual feeding each option can be tailored to reduce cycle times.

Chapter 6 Conclusion

6.1 General conclusion

Panelized construction has proven to be a promising method due to its use of an off-site manufacturing philosophy, which provides a higher quality product with reduced site disruptions and a shorter construction cycle. However, among the manufacturing processes in panelized construction, there is reduceable waste produced by machine cutting of the building materials. As such, there is an opportunity to better realized the benefits of the off-site philosophy by automating various light-framed building processes. In order to advance the current advantages of panelized construction, this research investigates automated solutions for reducing waste and prototype machine designs for wood and light-gauge steel framing machines in order to facilitate increased automation within the panelized construction.

This research begins by developing an optimization model for maximizing the utilization of raw stock during the automatic cutting of floor components based on dynamic waste allocation. The prototype system is able to reduce cost and waste by reallocating potential unused material to less critical components of the floor structure. The proposed model can extract and analyze data from multiple parallel jobs, allowing for accurate prediction of the required stock and resulting waste before the commencement of the actual cutting process. Finally, the proposed solution can be integrated into other related cutting applications, where the design allows for reallocation of waste from other components. Three case studies from an Edmonton-based panelized manufacturer are presented to demonstrate the accuracy and efficiency of implementing the proposed model.

Second, a prototype machine currently being tested at the University of Alberta, which will support the industrialization of off-site construction by assisting in assembling and fastening of light-gauge steel framed wall-panels, is evaluated. The proposed model is used for transferring manufacturing information outputted from BIM models to an open-source file format readable by the programmable logic controller (PLC) used to control the steel framing machine prototype. The proposed model allows for the generation of collision-free tool trajectories through the implementation of path generation, and collision detection and avoidance modules. These modules are designed to intelligently plan operations for safe tool movements and easy integration of in-shop modifications. Real-world results from various panel configurations are shown to validate the accurate information transfer between the 3D-BIM outputs and the prototype machine.

Lastly, this thesis describes the development of simulation models for the automated light gauge steel framing process using discrete-event simulation (DES) mimicking real-time machine production capacity and cycle time. At present, the literature on the development of such models for automated construction machinery is lacking; in this context, this research aims to showcase the advantages of simulation as a decision-making support tool. The development of these models provides a useful tool for understanding bottlenecks in machine operations that can be addressed to meet local demand. Since the existing steel framing process primarily consists of manual assembly and fastening of cold-formed steel (CFS) frames, these models showcase the potential to increase the level of automation through the addition of various mechanical and control modifications to an existing prototype steel framing machine.

6.2 Research contributions

The contributions of this research can be summarized as follows:

1. A heuristic cutting approach for reducing the material waste by reallocating otherwise wasted material to less critical components of the floor structure (Objective 1).
2. A framework for transferring manufacturing information outputted from BIM models to an open-source file format readable by the PLC (Objective 2).
3. A methodology for generation of collision-free tool trajectories through the implementation of collision detection and avoidance modules (Objective 3).
4. A methodology for generation of multi-spindle path planning for optimal execution of wood framing processes, including nailing, cutting, and drilling (Objective 3).
5. Application of throughput time reduction techniques to support more thorough understanding of the manufacturing capacity and manufacturing time for automated construction machinery (Objective 4).

6.3 Research limitations

This research is subject to the following limitations:

1. The methodology for collision-free tool paths generation lacks the capability of generating safe tool paths for steel panels when the working time in one buffer zone is higher than the cumulative time in adjacent working and buffer zones.
2. In this research, tool-path generation for steel framing machine is limited to coupled cases, where for every screw-fastening operation a top and a bottom

operation must be present. This coupling in screw-fastening operation limits the path generation to only frames composed of C-shaped metal studs, excluding the integration of special studs such as metal wall angle studs.

3. Both wood and steel framing machines are designed for rectangular panels; thus, the proposed research is limited to the production of such configurations.
4. A heuristic cutting approach for minimizing floor component cutting waste is highly effective when executing multiple floor plans in parallel; however, the potential savings may be offset by increased complexity in sorting the cut pieces.

6.4 Future research

The research presented serves as a foundation for further automation of panel manufacturing (see Figure 6.1). The following areas require further research:

1. The addition of machine-vision technology will validate the CAM output generated from the BIM model. In turn, the acquisition of real-world data will allow for the adjustment of pre-existing tool-paths using MMRW framework.
2. Real-time quality control for screw-fastening operation will ensure panel construction in accordance with the BIM models; moreover, implementation of such systems will provide intelligent integration of as-built information into BIM models for real-time progress and quality monitoring.
3. The investigation of a material sorting system to follow the cutting operation of floor components will allow for more significant material savings when executing multiple jobs in parallel.

4. Further improvements to path planning to account for various panel configurations (e.g., diagonal bracings, L-shaped studs, and dimpled studs) will broaden the applicability of the proposed algorithms and machines.
5. Integration of different material thicknesses (e.g., 22-, 18-, 16-, and 12-gauge steel) to account for screw-fastening time will further enhance the collision detection and avoidance modules and will provide greater versatility for manufacturing of panels of varying design configurations.

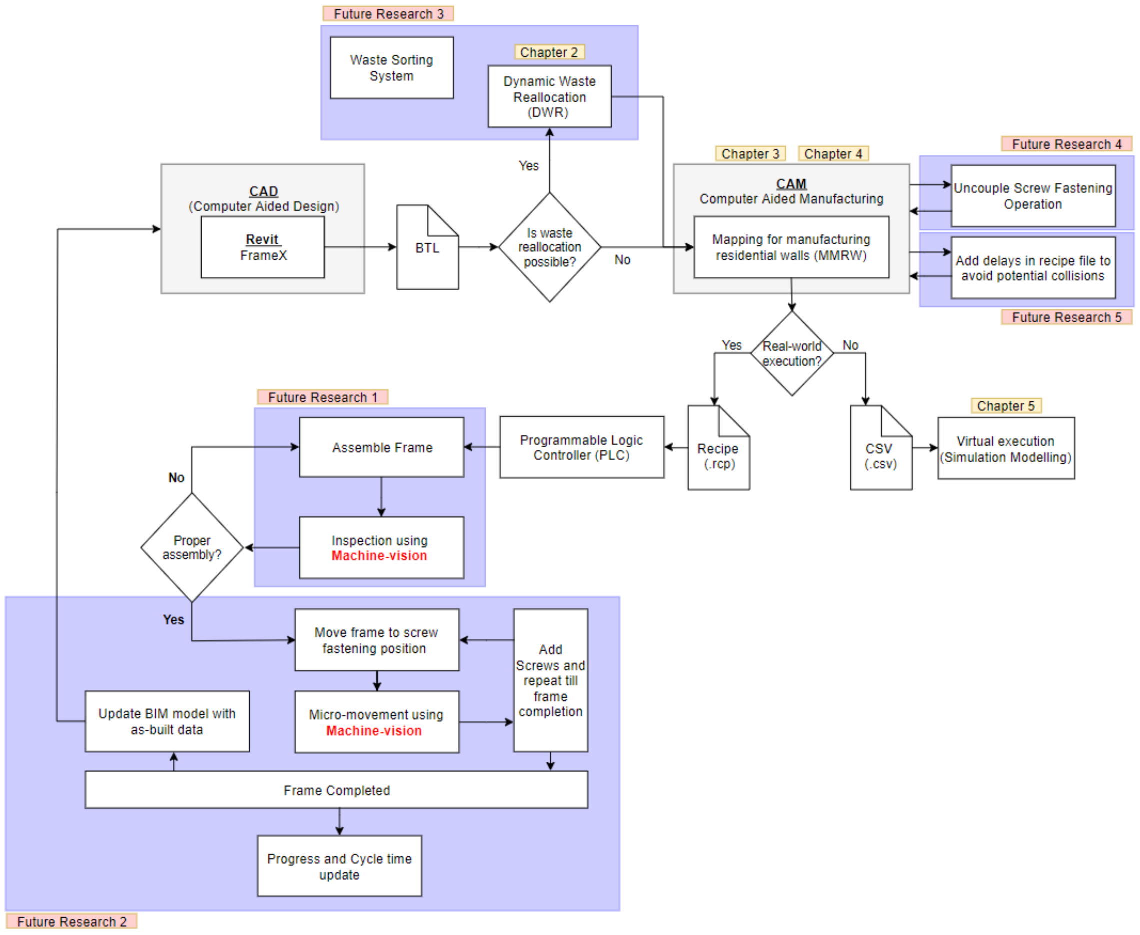

Future Research 3
Waste Sorting System
Chapter 2
Dynamic Waste Reallocation (DWR)
Yes
Is waste reallocation possible?
No
CAD
(Computer Aided Design)
Revit
FrameX
BTL
Chapter 3
Chapter 4
CAM
Computer Aided Manufacturing
Mapping for manufacturing residential walls (MMRW)
Future Research 4
Uncouple Screw Fastening Operation
Add delays in recipe file to avoid potential collisions
Future Research 5
Real-world execution?
Yes
No
Recipe (.rcp)
CSV (.csv)
Chapter 5
Virtual execution (Simulation Modelling)
Programmable Logic Controller (PLC)
Future Research 1
Assemble Frame
Inspection using Machine-vision
No
Proper assembly?
Yes
Move frame to screw fastening position
Add Screws and repeat till frame completion
Micro-movement using Machine-vision
Update BIM model with as-built data
Frame Completed
Progress and Cycle time update
Future Research 2

Figure 6.1 Research contributions and proposed future research

References

Afifi, M., Al-Hussein, M., AbouRizk, S., Fotouh, A., & Bouferguene, A. (2016). Discrete and Continuous Simulation Approach to Optimize the Productivity of Modular Construction Element. In *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction* (Vol. 33, p. 1). Vilnius Gediminas Technical University, Department of Construction Economics & Property.

Aguair, M. L. & Behdinan, K. (2015). Design, Prototyping, and Programming of a Bricklaying Robot. *Journal of Student Science and Technology*, *8*(3).

Ahmad, R. & Plapper, P. (2015). Generation of Safe Tool-path for 2.5 D Milling/Drilling Machine-tool using 3D ToF Sensor. *CIRP Journal of Manufacturing Science and Technology*, *10*, 84–91.

Ahmad, R., Tichadou, S., & Hascoet, J.-Y. (2016). Generation of safe and intelligent tool-paths for multi-axis machine-tools in a dynamic 2D virtual environment. *International Journal of Computer Integrated Manufacturing*, *29*(9), 982–995.

Ahmad, R., Tichadou, S., & Hascoet, J. Y. (2017). A knowledge-based intelligent decision system for production planning. *International Journal of Advanced Manufacturing Technology*, *89*(5–8), 1717–1729. http://doi.org/10.1007/s00170-016-9214-z

Aldana, J. C. & Serpell, A. (2013). Construction Waste Management System : A Case Study in A Construction Project In Chile. *Proceedings of the 19th CIB World Building Congress, May 2013*, *15*(2010), 32–41.

Allen, E. & Iano, J. (2009). *Fundamentals of Building Construction: Materials and Methods* (5th ed.). Hoboken, NJ: John Wiley & Sons.

Altaf, M. S., Al-Hussein, M., & Yu, H. (2014). Wood-frame wall panel sequencing based on discrete-event simulation and particle swarm optimization. In *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction* (Vol. 31, p. 1). Vilnius Gediminas Technical University, Department of Construction Economics & Property.

Altaf, M. S., Liu, H., Al-Hussein, M., & Yu, H. (2015). Online simulation modeling of prefabricated wall panel production using RFID system. In *Winter Simulation Conference (WSC), 2015* (pp. 3379–3390). IEEE.

Avdzhieva, A., Balabanov, T., Evtimov, G., Kirova, D., Kostadinov, H., Tsachev, T., … Zlateva, N. (2014). *Optimal Cutting Problem*. Retrieved from http://www.maths-in-industry.org/miis/692/

Azab, A. & AlGeddawy, T. (2012). Simulation methods for changeable manufacturing. *Procedia CIRP*, *3*, 179–184.

Bock, T. (2015). The future of construction automation: Technological disruption and the upcoming ubiquity of robotics. *Automation in Construction*, *59*(Supplement C), 113–121. http://doi.org/https://doi.org/10.1016/j.autcon.2015.07.022

Bock, T. & Linner, T. (2016). *Construction robots : Elementary technologies and single-task construction robots.* New York, NY: Cambridge University Press, 2016. http://doi.org/9781107075993

Burdett, R. L. & Kozan, E. (2014). An integrated approach for earthwork allocation, sequencing and routing. *European Journal of Operational Research*, *238*(3), 741–759.

Carballo, E. A. S., Morales, L., & Trujillo-Romero, F. (2017). Path Planning for a Mobile Robot using Genetic Algorithm and Artificial Bee Colony. In *2017 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)* (pp. 8–12). http://doi.org/10.1109/ICMEAE.2017.16

Chen, H.-C., Yau, H.-T., & Lin, C.-C. (2012). Computer-aided process planning for NC tool path generation of complex shoe molds. *International Journal of Advanced Manufacturing Technology*, *58*(5–8), 607–619. Retrieved from http://10.0.3.239/s00170-011-3398-z

Cheng, C., Feiring, B., & Cheng, T. C. (1994). The cutting stock problem: A survey. *International Journal of Production Economics*, *36*(3), 291–305. http://doi.org/10.1016/0925-5273(94)00045-X

Cherri, A. C., Arenales, M. N., & Yanasse, H. H. (2013). The usable leftover one-dimensional cutting stock problem-a priority-in-use heuristic. *International Transactions in Operational Research*, *20*(2), 189–199. http://doi.org/10.1111/j.1475-3995.2012.00868.x

Chu, B., Jung, K., Lim, M.-T., & Hong, D. (2013). Robot-based construction automation: An application to steel beam assembly (Part I). *Automation in Construction*, *32*, 46–61. http://doi.org/https://doi.org/10.1016/j.autcon.2012.12.016

CMHC's Housing Market Outlook (2017). *Housing Market Outlook — Canada and Major Centres.* Retrieved from <https://www.cmhc-schl.gc.ca/en/data-and-research/publications-and-reports/housing-market-outlook-canada-and-major-centres > (accessed Oct. 02, 2018).

Cui, Y. & Yang, Y. (2010). A heuristic for the one-dimensional cutting stock problem with usable leftover. *European Journal of Operational Research*, *204*(2), 245–250. http://doi.org/10.1016/j.ejor.2009.10.028

Davies, J. M. (2006). Light gauge steel cassette wall construction—theory and practice. *Journal of Constructional Steel Research*, *62*(11), 1077–1086.

design2machine. (2016). *Documentation: btl interface description.* Retrieved from <https://design2machine.com/btl/> (accessed Dec. 31, 2018).

Dikili, A. C., Sarıöz, E., & Pek, N. A. (2007). A successive elimination method for one-dimensional stock cutting problems in ship production. *Ocean Engineering*, *34*(13), 1841–1849.

Fayzrakhmanov, R. A., Murzakaev, R. T., Mezentsev, A. S., & Shilov, V. S. (2014). Applying the greedy algorithm for reducing the dimensionality of the dynamic programming method in solving the one-dimensional cutting stock problem. *Middle-East Journal of Scientific Research*, *3*(19), 412–416.

Fiorino, L., Della Corte, G., & Landolfo, R. (2007). Experimental tests on typical screw connections for cold-formed steel housing. *Engineering Structures*, *29*, 1761–1773. Retrieved from http://10.0.3.248/j.engstruct.2006.09.006

Garraffa, M., Salassa, F., Vancroonenburg, W., Vanden Berghe, G., & Wauters, T. (2016). The one-dimensional cutting stock problem with sequence-dependent cut losses. *International Transactions in Operational Research*, *23*(1–2), 5–24.

Gilmore, P. C. & Gomory, R. E. (1961). A Linear Programming Approach to the Cutting-Stock Problem. *Operations Research*, *9*(6), 849–859. http://doi.org/10.1287/opre.9.6.849

Gündüz, M., Kiran, M. S., & Özceylan, E. (2015). A hierarchic approach based on swarm intelligence to solve the traveling salesman problem. *Turkish Journal of Electrical Engineering & Computer Sciences*, *23*(1), 103–117.

Gupta, R. (2011). The Innovative Precast Light Gauge Steel Wall System Combined with Cellular Concrete (pp. 8–10).

Gutin, G. & Punnen, A. P. (2006). *The traveling salesman problem and its variations* (Vol. 12). Springer Science & Business Media.

Herrmann, J. W. (2003). Design for production: Concepts and applications. *Proceedings of the SME East Coast Region*, *3*, 1–9.

Herrmann, J. W. & Chincholkar, M. M. (2000). Design for production: A tool for reducing manufacturing cycle time. In *Proceedings of the 2000 ASME Design Engineering Technical Conference* (pp. 10–13).

Hinterding, R. & Khan, L. (1995). Genetic algorithms for cutting stock problems: With and without contiguity. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol.

956, pp. 166–186). Springer, Berlin, Heidelberg. http://doi.org/10.1007/3-540-60154-6_54

Jahangirian, M., Eldabi, T., Naseer, A., Stergioulas, L. K., & Young, T. (2010). Simulation in manufacturing and business: A review. *European Journal of Operational Research*, *203*(1), 1–13.

Johnson, D. J. (2003). A framework for reducing manufacturing throughput time. *Journal of Manufacturing Systems*, *22*(4), 283.

Kizilateş, G. & Nuriyeva, F. (2013). On the nearest neighbor algorithms for the traveling salesman problem. In *Advances in Computational Science, Engineering and Information Technology* (pp. 111–118). Springer.

Lavoie, P., Kenné, J.-P., & Gharbi, A. (2007). Production control and combined discrete/continuous simulation modeling in failure-prone transfer lines. *International Journal of Production Research*, *45*(24), 5667–5685.

Lawler, E. L. (1985). *The Traveling Salesman Problem : A Guided Tour of Combinatorial Optimization*. Chichester [West Sussex]: Wiley, Hoboken, NJ.

Lawson, R. M. & Ogden, R. G. (2008). "Hybrid" light steel panel and modular systems. *Thin-Walled Structures*, *46*(7), 720–730. http://doi.org/https://doi.org/10.1016/j.tws.2008.01.042

Lee, Y. H., Cho, M. K., Kim, S. J., & Kim, Y. B. (2002). Supply chain simulation with discrete–continuous combined modeling. *Computers & Industrial Engineering*, *43*(1–2), 375–392.

Li, X. (2016). Process automation for flexible residential wall panel manufacturing. University of Alberta.

Liang, K.-H., Yao, X., Newton, C., & Hoffman, D. (1998). Solving cutting stock problems by evolutionary programming. In V. W. Porto, N. Saravanan, D. Waagen, & A. E. Eiben (Eds.), *Evolutionary Programming VII: 7th International Conference, EP98 San Diego, California, USA, March 25–27, 1998 Proceedings* (pp. 755–764). Berlin, Heidelberg: Springer Berlin Heidelberg. http://doi.org/10.1007/BFb0040826

Liu, H. (2016). *BIM-based Automated Planning for Panelized Construction in the Light-Frame Building Industry*. *doi.org*. University of Alberta Libraries. Retrieved from https://era-library-ualberta-ca.login.ezproxy.library.ualberta.ca/files/cj38607172#.WYdgbIjythE

Liu, H., Altaf, M. S., Lei, Z., Lu, M., & Al-Hussein, M. (2015). Automated production planning in panelized construction enabled by integrating discrete-event simulation and BIM.

Liu, X., Zhang, A., Ma, J., Tan, Y., & Bai, Y. (2015). Design and model test of a modularized prefabricated steel frame structure with inclined braces. *Advances in Materials Science and Engineering*, *2015*.

Malik, N., Ahmad, R., & Al-Hussein, M. (2018). Investigating effects of reduced technological constraints on cycle time through simulation modeling for automated steel wall framing. In *Modular and Offsite Construction Summit*. Hollywood, FL, USA.

Manrique, J. D., Al-Hussein, M., Bouferguene, A., & Nasseri, R. (2015). Automated generation of shop drawings in residential construction. *Automation in Construction*, *55*, 15–24.

Negahban, A. & Smith, J. S. (2014). Simulation for manufacturing system design and operation: Literature review and analysis. *Journal of Manufacturing Systems*, *33*(2), 241–261.

Oldcastle Business Intelligence (2017). *2018 North American Construction Forecast Report.* Retrieved from <https://info.buildingsolutions.com/hubfs/2018%20North%20American%20Construction%20Forecast%20Report.pdf?t=1514916335905> (accessed Oct. 02, 2018).

Ossimitz, G. & Mrotzek, M. (2008). The basics of system dynamics: Discrete vs. continuous modelling of time. In *The 26th Conference of SD*. Citeseer.

Paudel, P., Dulal, S., Bhandari, M., & Tomar, A. (2016). Study on Pre-fabricated Modular and Steel Structures. *SSRG International Journal of Civil Engineering (SSRG-IJCE)*, *3*(5), 15.

Peña-Mora, F., Han, S., Lee, S., & Park, M. (2008). Strategic-operational construction management: Hybrid system dynamics and discrete event approach. *Journal of Construction Engineering and Management*, *134*(9), 701–710.

Rabello Golfeto, R., Moretti, A. C., & Salles Neto, L. L. de. (2008). A grasp metaheuristic for the ordered cutting stock problem. *Ingeniare. Revista Chilena de Ingeniería*, *16*(3).

Ragsdale, C. T. & Zobel, C. W. (2004). The ordered cutting stock problem. *Decision Sciences*. Blackwell Publishing, Inc. http://doi.org/10.1111/j.1540-5414.2004.02505.x

Ren, L., Sparks, T., Ruan, J., & Liou, F. (2010). Integrated process planning for a multiaxis hybrid manufacturing system. *Journal of Manufacturing Science and Engineering*, *132*(2), 21006.

Rui, L. I. U. (2017). Comparison of Evolutionary Computing Algorithms for Solving Robot Path Planning. *Revista de La Facultad de Ingeniería*, *32*(7).

Shahzad, W., Mbachu, J., & Domingo, N. (2015). Marginal productivity gained through prefabrication: Case studies of building projects in Auckland. *Buildings*, *5*(1), 196–208.

Shen, X., Xie, F., Liu, X.-J., & Ahmad, R. (2017). An NC Code Based Machining Movement Simulation Method for a Parallel Robotic Machine. In *International Conference on Intelligent Robotics and Applications* (pp. 3–13). Springer.

Singh, R. P. (2014). Application of Graph Theory in Computer Science and Engineering. *International Journal of Computer Applications*, *104*(1).

Sinuany-Stern, Z. & Weiner, I. (1994). The one dimensional cutting stock problem using two objectives. *Journal of the Operational Research Society*, *45*(2), 231–236.

Statistics Canada (2018). *Table 34-10-0135-01 Canada Mortgage and Housing Corporation, housing starts, under construction and completions, all areas, quarterly*. Retrieved from

<https://www150.statcan.gc.ca/t1/tbl1/en/cv.action?pid=3410013501#timeframe > (accessed Oct. 02, 2018).

Tam, C. M., Tam, V. W. Y., Chan, J. K. W., & Ng, W. C. Y. (2005). Use of Prefabrication to Minimize Construction Waste: A Case Study Approach. *International Journal of Construction Management*, *5*(1), 91–101. http://doi.org/10.1080/15623599.2005.10773069

Tam, V. W. Y., Tam, C. M., Chan, J. K. W., & Ng, W. C. Y. (2006). Cutting Construction Wastes by Prefabrication. *International Journal of Construction Management*, *6*(1), 15–25. http://doi.org/10.1080/15623599.2006.10773079

Tamayo, E., Bardwell, M., Qureshi, A., & Al-Hussein, M. (2017). Automation of a steel wall framing assembly. In *International Structural Engineering & Construction Conference (ISEC)* (pp. 24–29). Valencia, Spain.

Usman, M. (2017). *Predictive Analysis on Large Data for Actionable Knowledge: Emerging Research and Opportunities*. Information Science Reference. Retrieved from https://books.google.ca/books?id=z3XdtAEACAAJ

Wang, F.-K. & Liu, F.-T. (2014). Flexible stock allocation and trim loss control for cutting problem in the industrial-use paper production. *Mathematical Problems in Engineering*, *2014*.

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, *183*(3), 1109–1130. http://doi.org/10.1016/j.ejor.2005.12.047

Weiss, A. (2005, February 15). System and method of panelized construction. U.S. Patent No. 6,854,218.

Weyerhaeuser (2018). *Featuring Trus Joist® TJI® Joists for Floor and Roof Applications*. Retrieved from <https://www.weyerhaeuser.com/application/files/5015/3417/5537/TJ-4000.pdf> (accessed Oct. 02, 2018).

Xu, X. W. & Newman, S. T. (2006). Making CNC machine tools more open, interoperable and intelligent—a review of the technologies. *Computers in Industry*, *57*(2), 141–152.

Yeheyis, M., Hewage, K., Alam, M. S., Eskicioglu, C., & Sadiq, R. (2013). An overview of construction and demolition waste management in Canada: A lifecycle analysis approach to sustainability. *Clean Technologies and Environmental Policy*, *15*(1), 81–91.

Yu, W. & LaBoube, R. A. (2010). *Cold-formed steel design* (4th ed.). Hoboken, NJ: Wiley. ISBN: 978-0-470-46245-4

Yusof, Y. & Latif, K. (2015). A novel ISO 6983 interpreter for open architecture CNC systems. *International Journal of Advance Manufacturing Technology*.

Zhang, J. & Khoshnevis, B. (2013). Optimal machine operation planning for construction by Contour Crafting. *Automation in Construction*, *29*(Supplement C), 50–67. http://doi.org/https://doi.org/10.1016/j.autcon.2012.08.006

Zhao, H. (2015). *Automation of Quantity Takeoff and Material Optimization for*

Residential Construction Manufacturing. M.Sc., University of Alberta, 2015