Optimization of A Real Time Multi Mixed Make-To-Order Assembly Line to Reduce Positive Drift

By:

RANGITH BABY KURIAKOSE

Thesis submitted in fulfilment of the requirements for the Degree

DOCTOR OF ENGINEERING: ENGINEERING: ELECTRICAL

in the

Department of Electrical, Electronic and Computer Engineering

of the

Faculty of Engineering Built Environment and Information Technology

at the

Central University of Technology, Free State

Supervisor: Prof HJ Vermaak (Ph. D)

BLOEMFONTEIN

2019

Central University of Technology, Free State

DECLARATION OF INDEPENDENT WORK

I, RANGITH BABY KURIAKOSE, identity number and student number , do hereby declare that this research project submitted to the Central University of Technology, Free State for the Degree DOCTOR OF ENGINEERNG: ELECTRICAL ENGENIEERING, is my own independent work; and complies with the Code of Academic Integrity, as well as other relevant policies, procedures, rules and regulations of the Central University of Technology, Free State; and has not been submitted before to any institution by myself of any other person in fulfilment (or partial fulfilment) of the requirements for the attainment of any qualification.

[illegible]

SIGNATURE OF STUDENT **DATE**

Dedication

To my Parents, Baby Kuriakose and Lissy Baby, for dedicating their time, effort and prayers to me at all times.

Acknowledgements

I wish to extend my sincere gratitude to the following for their contribution to the completion of this study.

Firstly, to God Almighty for giving me the strength, perseverance and patience to complete this study.

To Professor Herman Jacobus Vermaak, my supervisor, I am grateful for the undivided attention, guidance and motivation you provided throughout this study. Without your support, I could not have completed this study.

I would like to thank the Central University of Technology, Free State (CUT) for providing me with financial support to attend workshops and conferences that assisted me in gaining further knowledge on the study I embarked on.

I would like to thank the Research Group in Evolvable Manumation Systems (RGEMS) and my departmental colleagues, especially Professor Kanzumba Kusakana, Dr Benjamin Kotze and Mr. Gareth Gericke for providing the structural support that was critical throughout this study.

I would like to thank my Parents, Mr. Baby Kuriakose and Mrs. Lissy Baby, my Sister Elizabeth and my Brother in Law Ajay. Your words of advice, encouragement and motivation went a long way in helping me completing this study. I also remember with great gratitude my uncles', the late Professor PM George and Professor PM Kuriakose, who mentored me during my formative days.

Finally, I would like to thank in a special way my beloved wife, Reenu Thomas and my loving daughter Olivia. Your words of motivation and prayers helped me get through some very challenging tests throughout this journey. Without your presence by my side, I could not have completed this effort. Thank you.

Abstract

Assembly lines are critical for the realization of product manufacture. In recent times, there has been a shift from the make-to-stock (mass production) approach to a make-to-order (mass customization) approach and this has brought on a strong emphasis on product variety. Although variety can be included to a product at various phases of production, literature shows that by providing each functional module of the product with several variants, assembly lines provide the most cost-effective approach to achieve high product variety. However, there are certain challenges associated with using assembly lines to achieve product variety. One of these challenges is assembly line balancing. Assembly line balancing is the search for an optimum assignment of tasks, such that given precedence constraints according to pre-defined single or multi objective goal are met. These objectives include reducing the number of stations for a given cycle time or minimizing the cycle time for a given number of stations. Cycle time refers to the amount of time allotted to accomplish a certain process in an assembly process. This deviation from the optimal cycle time is technically referred to as drift. Drift can be negative or positive. Negative drift represents the time span during which an assembly line is idle, due to work being finished ahead of prescribed cycle time. Positive drift, meanwhile, represents time span in which an assembly line exceeds the prescribed cycle time. The problems caused by drift, especially positive drift, is so vast that there is a research niche are dedicated to this study called Assembly Line Balancing Problems. Various authors have proposed numerous solutions for solving assembly line balancing problems created by positive drift. However, there is very little information on optimizing multi model make-to order systems with real time inputs so as to reduce the effects of positive drift. This study looks at how such a system can be optimized by using the case study of a water bottling plant. This is done by initially looking at the literature in the field of assembly line balancing to isolate the research gap this study aims to fill. Secondly, the water bottling plant, described as the case study, is modelled using MATLAB/Simulink. Thirdly, the different optimization methodologies are discussed and applied to the created model. Finally, the optimized model is tested and the results are analysed. The results of this study show that positive drift, which can be a major challenge in a real time multi mixed assembly line, can be reduced by the optimization of assembly lines. The results of this study can also be seen as an addition to the knowledge base of the broader research on mixed model assembly line balancing.

Contents

List of Figures

List of Tables

Chapter 1 Introduction to study environment

1.1 Introduction

Traditionally, industries have focussed on producing a limited variety of products by stocking raw materials ahead of demand and shipping in accordance with demand. This approach is termed as make-to-stock [1]. The main risk associated with the make-to-stock approach is that product demand is stochastic [2].

In order to overcome this obvious disadvantage, most industries now opt for the make-to-order approach [3]. A make-to-order assembly line is one which has the ability to fulfil [4] customer's choice in adding variety to the products. The major advantages of such an approach is that it eliminates finished inventories that may remain unsold, reduces financial risk and increases product variety [4].

However, as product variety increases, due to the shift from mass production to mass customization [5], assembly lines must be designed to adapt accordingly and operate to meet the demands of product variety. Product variety [6] can be included to a product at various phases of production. It can be included [5] in the design, fabrication, assembly, sales or use phase of product manufacture. In design phase, a product can be designed to include preferences of a specific user deeming it a personalized or designer product. In the fabrication process, a product can be manufactured using 3-D printers, rapid prototyping or machining depending on cost, time and user specifications. Variety can also be introduced in the sales and use phase of certain product, as they can be tailored for use in accordance with personal preferences.

The assembly line however, provides one of the most cost effective approaches to high product variety [5]. This is achieved by providing each functional module of the product with several variants, so that the assembly combination will provide high variety in the final products. The economy of scope of high variety is achieved in the final assembly by using reconfigurable systems.

However, high variety in products and quick response time to execute product manufacture is often conflicting [7]. One of the obstacles in achieving this quick response times is assembly line balancing [8]. Assembly line balancing is the search for the optimum assignment of tasks to stations given precedence constraints according to pre-defined single or multi objective goal.

These objectives [5] include reducing the number of stations for a given cycle time, minimizing the cycle time for a given fixed number of stations and to minimize the smoothness index for a given number of stations. When these objectives are not met, they give rise to assembly line balancing problem(ALBP) [9].

The assembly line balancing problem specific to the minimization of cycle time for allocating tasks to a fixed number of workstations is classified as ALBP-2 [10]. Cycle time refers to the amount of time allotted to accomplish a certain process in an assembly process. Minimizing cycle time [11], [12] is comparatively easier if a single product is manufactured in an assembly line as opposed to multiple products [13].

A factor affecting cycle time is drift. Drift refers to the deviation from the optimal cycle time. Drift can be negative or positive. Negative drift represents the time span during which an assembly line is idle, due to work finished ahead of prescribed cycle time or awaiting components. Positive drift, meanwhile, represents time span in which as assembly line exceeds the prescribed cycle time [14].

Several studies have noted the severity of challenges that positive drift brings to the order of scheduling in an assembly line [15][16]. Specific studies have also further noted the adverse effects of positive drift to the order of scheduling of make-to-order systems [17]. It is key to note that most of these studies are on systems which have a fixed routine. The natural assumption in this case must be that the challenges posed by positive drift on make-to-order systems will be much more severe as several models are manufactured on the same assembly line.

The area of research which focuses on the balancing of assembly lines which produce multiple models of a product is called Multi/Mixed Model Assembly Line Balancing (MMALB) [18].However, there is seemingly a lack of an in-depth study on how to balance real time mixed model assembly line therefore it deems an in-depth study. Real time optimization is critical for the successful planning of a diverse assembly line [19]. It would consider factors like production rate and production time interval among others to ensure quick decisions to aid the make-to-order [4] approach. This research aims to construct a mathematical model for a mixed model assembly line that accepts real time inputs and optimize the cycle time such that least time in product manufacture is achieved thereby reducing positive drift.

1.2 Problem statement

As competition in high tech markets increase, product differentiation and customization become top priority. This has resulted in most assembly lines switching from make-to stock approach to a make-to-order approach. However, positive drift remains a major challenge to the cost-effective operation of any assembly line.

1.3 Research hypothesis and objectives

1.3.1 Research hypothesis

An optimized real time multi mixed model assembly line can reduce the effects of positive drift thereby allowing for assembly lines to function more efficiently, increase productivity and product variety.

1.3.2 Research objectives

The optimization of a multi mixed assembly line poses a few challenges. The first and foremost of which is that experimentation cannot be done on an existing mixed model assembly line as it might result in disruption and delay in productivity. In addition to this challenge, it is difficult to add real time inputs to an existing plant as they operate on a fixed routine. To overcome these and other challenges the following objectives are set forth in this study;

- Study the broader problem of assembly line balancing and then focus where the challenge described in this research fits in thereby establishing the research gap this study aims to fill.
- Using a case study, develop a mixed model assembly line that can be used to conduct experiments.
- Design a user interface so that the model can accept real time inputs
- Simulate the model for various inputs and record the respective cycle time.
- Formulate the objective functions, main constraints and equality constraints to optimize the model.
- Test the veracity of the optimized model using random inputs and constraints.

1.4 Research methodology

As alluded to in the problem statement, positive drift poses a major challenge to the cost-effective operation of an assembly line. This challenge is accentuated when a real time make to order assembly line, such as the one studied in this research, is considered. The aim of the study is to mainly develop an optimization model that will contribute to the research niche area of Multi/Mixed Model Assembly Line Balancing Problem (MMALBP) and to create a physical system based on the model on which study can be done in future.

In order to meet this challenge a case study of a water bottling plant is considered. The water bottling plant will need to manufacture and fill 500ml and 750ml bottles. The different sizes of the bottle make the system mixed model and introduce product variety. The real time inputs to the plant is provided using an external interface and can be provided from different points simultaneously.

The water bottling plant will be modelled using the MATLAB/Simulink software. A Cloud server will be set up to enable users to provide real time inputs to the plant Initially, real time simulations will be done on the model to see if it can produce the desired output of filling 500ml and 750ml bottles without affecting the inventory and at a manageable pace. This will ensure the rigour of the model.

A model optimization technique will be developed using the optimization toolbox of MATLAB/Simulink, which will accept the real time inputs provided to model using the Cloud interface. The optimization model will determine the best combination to yield maximum productivity while ensuring little or no positive drift.

The optimized model will then be tested with inventory constraints and multiple user inputs to see how it reacts to realistic assembly line conditions.

1.5 Thesis Layout

Chapter 1: The aim of Chapter 1 is to provide an overview of the research. In this chapter aspects such as the problem statement, hypothesis, research objectives and research methodology is introduced.

Chapter 2: The aim of Chapter 2 is to provide the literature study that was embarked upon prior to undertaking this study. This chapter initially gives an overview of Assembly Line systems and their classifications with a focus on multi mixed assembly lines, then it looks at Assembly Line Balancing Problems with a focus specifically on Positive Drift, thirdly, the focus is on the different optimization techniques used in Assembly line balancing and finally the limitations in the existing literature is examined.

Chapter 3: The aim of Chapter 3 is to showcase the research methodology used to develop the selected study. This chapter initially shows how a physical case study was selected to aid the study. It then goes to show how the case study was custom modelled in a MATLAB/Simulink environment. Next, the model is connected to a Cloud to enable real-time inputs and finally the model is optimized with the real-time inputs.

Chapter 4: The aim of this chapter is to detail the tests that were performed on the model at various stages of implementation. This is done by defining the model setup and the constraints that were used in each stage of the model development.

Chapter 5: The aim of this chapter is to discuss the results obtained from the tests conducted in the previous chapter. The chapter aims to provide an in-depth analysis of each stage of the model development and how it goes back to answer questions raised in the problem statement in Chapter 1.

Chapter 6: This chapter looks back at the work done in project and brings to the fore goals achieved during the project in terms adding knowledge to the specific field of research and the future scope of work.

Chapter 2 Literature Review

2.1 Introduction

The aim of this chapter is to look at the literature review that was undertaken prior to this study. The literature review will assist in establishing the research gap and justify the motivation for the study. The chapter is structured such that initially it focusses on introducing assembly line balancing. It then looks at Assembly Line Balancing Problems (ALBP). The chapter is rounded off by analyzing the Optimization techniques used in Assembly Line Balancing (ALB).

2.2 Assembly Line Balancing (ALB)

Assembly lines have been an integral component of the manufacturing scene from the very first industrial revolution to the latest and widely touted fourth industrial revolution, commonly referred to Industry 4.0 (I.4.0). The main aim of an assembly line has always been to [20] to ensure faster production by assigning tasks to different workstations positioned in serial production line.

Assembly lines have progressed significantly from the concept patent by Ransom E Olds [21] and the installation of the first moving assembly line [22] by Henry Ford. Assembly lines, today, are more dexterous and intelligent [23] owing to them being equipped with sensors that ensure a high degree of autonomy and less human interference [24].

However, irrespective of the assembly line being manual or automated, there has always been a decision process [12] that has been undertaken to assign tasks to a workstation to ensure maximum efficiency and least production time. This decision process is referred to as Assembly Line Balancing. There has been a consistent addition to the knowledge base of line balancing with the advent of new computing technologies and mathematical models.

This section aims to introduce the terminology and classification used in ALB. This will assist in understanding the various problems faced in line balancing.

2.2.1 Assembly Line Balancing Terminology

The terminology used in ALB is key to understanding the problems in line balancing. This section aims to define some of the terms and provide mathematical formulae for other.

- Assembly Line – A progressive production line [25] setup using a conveyor belt with workstations in regular intervals. Each workstation is assigned a specific task. After completion of the task the product is passed to the next station until the work is completed.

- Line balancing – The process by which the tasks assigned to an assembly line is levelled [3] so as to avoid bottle necks and excess capacity.
- Cycle time – The maximum time allowed to each workstation to complete an assigned task [26]. It is mathematically represented in equation (2.1)

$$Cycle\ time = \frac{Production\ time\ per\ day}{Units\ required\ per\ day} \qquad (2.1)$$

- Lead time – The total production time, taken as the sum of production time along the assembly line [3].
- Bottleneck – A delay in the production line caused by various factors [27] like unavailability of raw materials, slow workers and non-functioning machinery to name a few. Bottle neck is seen as one of the main reasons necessitating line balancing.
- Precedence – Defined as the order in which the product flows [28] through the different workstations in an assembly line. It can be a nodal or graphical representation.
- Drift – The deviation [29] from the optimal cycle time of a work station is defined as drift. Drift can be of two types, positive and negative. This section is elaborated in detail in Section 2.3.2.
- Smoothness Index – Defines the relative smoothness of operation of an assembly line. A perfectly balanced assembly line gives a zero smoothness index [3]. Smoothness index is mathematically represented in equation 2.2

$$SI = \sum_{j=1}^{k}(STmax - STj)^2 \qquad (2.2)$$

With,

SI = Smoothness index
k = number of workstations
STmax = Cycle time
STj = Time spent at station j

- Balance delay – The percentage of wasted time in an assembly line process. The mathematical representation [30] of balance delay is given in equation 2.3.

$$BD = \left(\frac{(k)*(CT)-(\sum_{j=1}^{k} STj)}{(k)*(CT)}\right) * 100\% \qquad (2.3)$$

With,

BD = Balance delay
k = number of workstations
CT = cycle time
STj = Time spent at station j

2.2.2 Classification of Assembly Line Balancing

This section aims to classify ALB. The classification is done according to the various parameters. Some of the most important parameters that need to be considered are described as follows;

- Product variety
- Task times
- Line layout
- Level of automation

sin

2.2.1.1 Product variety

Product variety, with respect to assembly line balancing, can be divided in three sections. They are detailed as follows

- Single model assembly lines – These are assembly lines that are designed to manufacture a single product [31]. Single model assembly lines are synonymous with mass production and are rare in today’s industry setup. An example of such an assembly line is one manufacturing compact discs [32].

- Mixed model assembly lines – These are assembly lines which are designed to include variety to the products. As a result, all products are variations of the same base product [33] and differ in specific aspects such as colour. The products are produced in a random mixed sequence hence might also cause a sequencing problem [34].
- Multi model assembly lines – These are models which are designed to produce products that are different [35] from each other. This would mean that the products cannot be manufactured in a sequence, like a mixed model assembly line. The products in a multi model assembly line need to be manufactured in batches with differing setup operations [33].

2.2.1.2 Task time

Task time, with respect to assembly lines, refer to the processing time taken by a certain work station to complete the task assigned to it. They are divided into two and are detailed as follows;

- Deterministic task times – These are assembly lines which have a predetermined processing time [36]. Although it is stated [37] that task times are never deterministic, they can be justified when the variation is negligibly small.
- Stochastic task times – These are assembly lines which do not assign a pre-determined processing time for a specific task. Factors [33] such as worker skill, work rate and availability of raw materials are often touted as reasons for stochastic task times.

2.2.1.3 Line layout

The Line layout refers to the organization of the assembly line. It can be generally classified into two. They are detailed as follows;

- Straight type (S-type) layout – This is the traditional layout of an assembly line where workstations are arranged along a straight line of the conveyor belt. This layout is suited [38] for a single model assembly line with a deterministic task time.
- U-type layout – The U-type assembly lines were born out of necessity as a result of change in the manufacturing setup. As the name suggests, the assembly line is organised in “U” shape [39]. In this arrangement, a worker is allowed to work on either side of the “U”. There is evidence that a U-type assembly line improves labour productivity [40].

2.2.1.4 Level of automation

The level of automation in an assembly line is divided into two main categories. They are described as follows;

- Manual line – As the name suggests, these are assembly lines which are operated manually. Contrary to popular belief, manual labour is still preferred in manufacturing industries where the work pieces are fragile and need to be gripped frequently [41].
- Automated lines – Automated assembly lines are mostly used in scenarios where the work environment is too hostile for human beings or where the labour costs are too high that manual labour is not economically viable [42].

2.3 Assembly Line Balancing Problems (ALBP)

The balancing of assembly lines has been defined as a decision process undertaken to assign tasks to a workstation to ensure maximum efficiency and least production time in Section 2.2. The problems encountered [12], [34], [43] when optimally dividing this decision process among the work stations with respect to an objective is referred to as an Assembly Line Balancing Problem (ALBP).

This section aims to firstly classify ALBP's, then in an aim to zoom in on the crux of this study, it will focus on Multi/Mixed Model Stochastic Systems (MMSS). Finally, this section sheds more light on positive drift which is a major problem within MMSS.

2.3.1 Classification of Assembly Line Balancing Problems

The Classification of ALBP has been closely examined in two [12], [44] major studies. For the purposes of this research, this section combines the two studies together to elaborate the classification. This section also analyses an ALBP classification study done by Sivasankaran and Shahabudeen [36] to bring out the specific research focus of this study and the apparent lack of research in that area.

Ghosh and Gagnon [44] did a classification of assembly lines based on the product variety (discussed in Section 2.2.1.1) and the task time (discussed in Section 2.2.1.2). This classification is shown in Figure 2.1

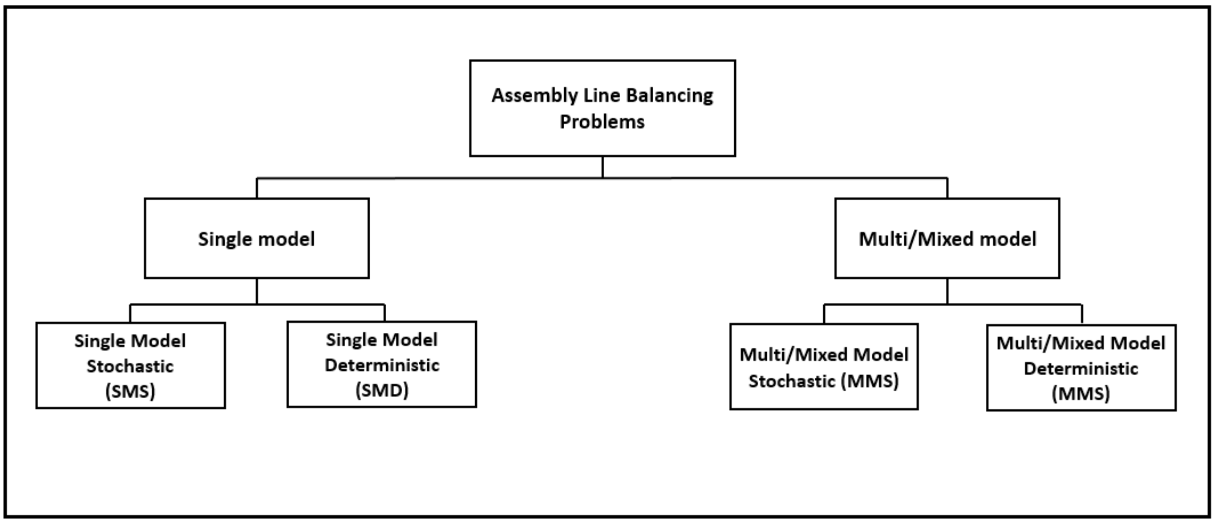

Assembly Line Balancing Problems
Single model
Multi/Mixed model
Single Model Stochastic (SMS)
Single Model Deterministic (SMD)
Multi/Mixed Model Stochastic (MMS)
Multi/Mixed Model Deterministic (MMS)

Figure 2. 1: ALBP classification according to Ghosh and Gagnon 1989 [44]

It can be seen from Figure 2.1 that the classification is split into single model and multi/mixed model assembly lines and then further split according to the task times. The task times can be stochastic (probabilistic) or deterministic.

Scholl and Becker [12], [34], then expanded the classification by Ghosh and Gagnon by adding sub-categories. This classification is shown in Figure 2.2.

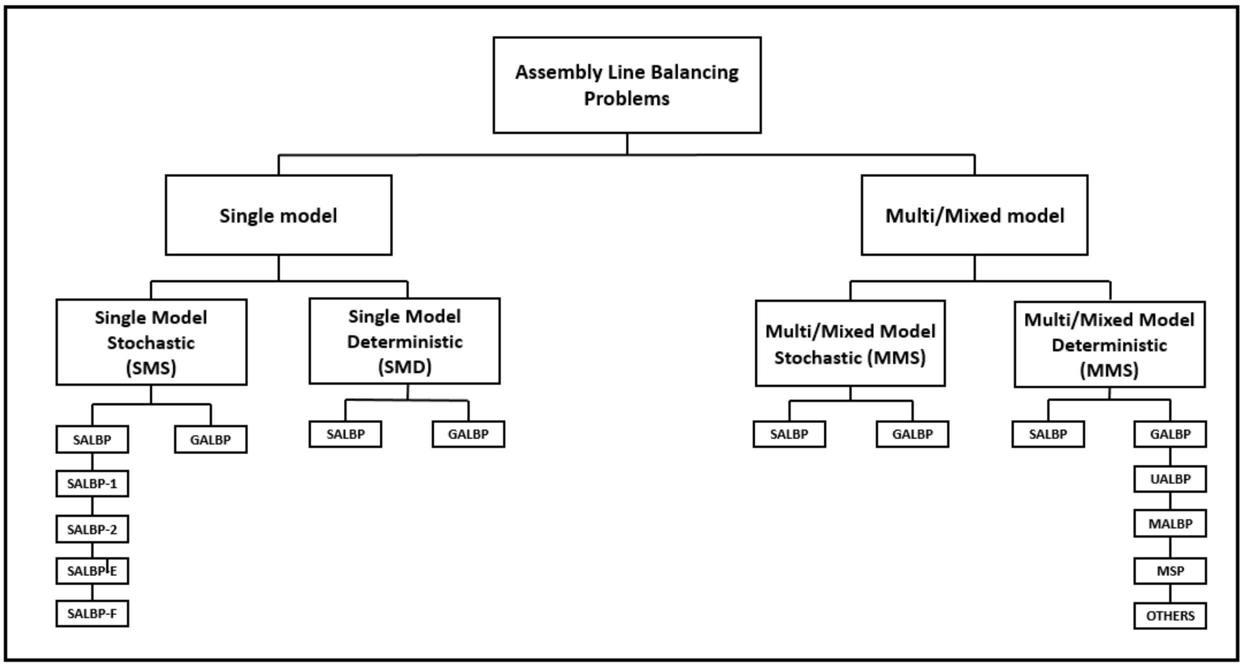

Assembly Line Balancing Problems
Single model
Multi/Mixed model
Single Model Stochastic (SMS)
Single Model Deterministic (SMD)
Multi/Mixed Model Stochastic (MMS)
Multi/Mixed Model Deterministic (MMS)
SALBP
GALBP
SALBP-1
SALBP-2
SALBP-E
SALBP-F
SALBP
GALBP
SALBP
GALBP
SALBP
GALBP
UALBP
MALBP
MSP
OTHERS

Figure 2. 2 ALBP classification according to Scholl and Becker 2006 [12]

The classification in Figure 2.2 shows the Single and Multi/Mixed model stochastic and deterministic assembly lines being split into the Simple Assembly Line Balancing Problems (SALBP) and Generalized Assembly Line Balancing Problems (GALBP). In the Single Model Stochastic assembly lines, the SALBP can be further split into four types [12], [28], [33]. They are listed as follows;

- SALBP-1 – Looking at problems concerned with reducing the number of workstations for a fixed cycle time.
- SALBP-2 – Looking at problems concerned with reducing the cycle time for a fixed number of workstations.
- SALBP-E – Looking at problems concerned with maximizing the efficiency (E) of the assembly line and thereby simultaneously minimizing cycle time and the workstations.
- SALBP-F – Looking at problems concerned with the feasibility of an assembly line for a given combination of cycle time and workstations.

All assembly line problems that fall outside the SALBP can be categorized [12] as GALBP. They can be further split three categories. They are listed as follows;

- MALBP – Looking at problems concerned with mixed model assembly lines (discussed in Section 2.2.1.1)
- MSP – Looking at problems concerned with mixed model sequencing such that factors like worker overlaid, line stoppage and off-line repair are addressed [45].
- UALBP – Looking at problems concerned with U shape line layout (discussed in Section 2.2.1.3) of an assembly line especially concerning the modified precedence constraints [46].

The main focus of the classification by Scholl and Becker [12] was on the Simple ALBP and that too on minimizing the cycle time and the number of workstations. There was some focus on balancing and sequencing problems of mixed model assembly lines, but this was categorised under Generalised ALBP.

Sivasankaran and Shahabudeen [36] picked up on this deficiency and progressed the research in a different direction by looking specifically at assembly line balancing problems of single and multi-model assembly with respect to their task time and line layout. Their classification is shown in Figure 2.3.

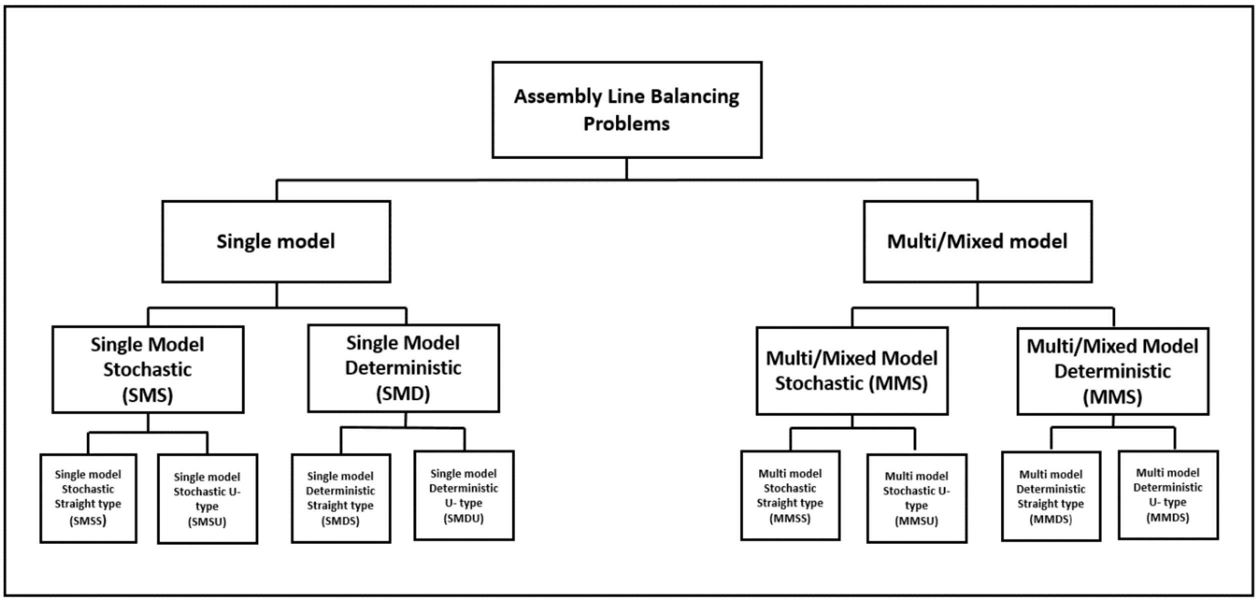

Assembly Line Balancing Problems
Single model
Multi/Mixed model
Single Model Stochastic (SMS)
Single Model Deterministic (SMD)
Multi/Mixed Model Stochastic (MMS)
Multi/Mixed Model Deterministic (MMS)
Single model Stochastic Straight type (SMSS)
Single model Stochastic U-type (SMSU)
Single model Deterministic Straight type (SMDS)
Single model Deterministic U- type (SMDU)
Multi model Stochastic Straight type (MMSS)
Multi model Stochastic U-type (MMSU)
Multi model Deterministic Straight type (MMDS)
Multi model Deterministic U- type (MMDS)

Figure 2. 3 ALBP classification by Sivasankaran and Shahabudeen 2014 [36]

As see from Figure 2.3, the classification by Sivasankaran and Shahabudeen [36] delves into the line layout of the assembly lines and their task time. This has been expanded below;

- Single Model Deterministic Straight type (SMDS) problem - The SMDS problem is the simplest form of all assembly line balancing problems. This type of a problem considers an assembly line manufacturing a single model in which the workstations are arranged in a straight line. The task time for each workstation is deterministic.
- Single Model Deterministic U-type (SMDU) problem - The SMDU problem considers the assembly of a single model as in the case of an SMDS problem. The difference is that the workstations are arranged in U form. The execution time for each workstation in the assembly line is deterministic.
- Single Model Stochastic Straight type (SMSS) problem - The SMSS problem considers the assembly of a single model as in the case of SMDS and SMDU problems. The difference is that the task times are stochastic and the work stations in the assembly line are in a straight line.
- Single Model Stochastic U- type (SMSU) problem - The SMSU problem considers the assembly of a single model as in the case of SMSS problems. The difference is that the

workstations are arranged in U form. As in the case of an SMSS problem, the task time for each workstation is stochastic.

- Multi Model Deterministic Straight type (MMDS) problem - The MMDS problem considers multiple models that need to be assembled in an assembly line. The work stations are aligned in a straight line and the task time for each model is deterministic. The mere fact that more than one product model is developed here means that the complexity of the MMDS problem is much more severe than single model deterministic problems.
- Multi Model Deterministic U-type (MMDU) problem - The MMDU problem differs from the MMDS problem in that the workstations are aligned in a U-type. Multiple models are developed here and the task time for the workstations for each model is deterministic.
- Multi Model Stochastic U-type (MMSU) problem - The MMSU problem is probably the most complex of the eight categories reviewed in this section. This is because this problem considers the assembly of multiple models in a U-type assembly line with each workstation having a stochastic task time.
- Multi Model Stochastic Straight type (MMSS) problem - The MMSS problem considers multiple models which need to be assembled in an assembly line with workstations aligned in a straight line and task times being stochastic. This makes it more complex as compared to a single model stochastic straight type (SMSS) problem.

2.3.2 Drift as a Problem in Assembly Line Balancing

Drift refers to the deviation from an optimal cycle time [5]. Drift can be positive or negative and is diagrammatically shown in Figure 2.4. As seen from Figure 2.4, positive drift occurs when work assigned to a station is not completed within the allotted cycle time (cycle time of Station A) and extends into the cycle time of the next work station (cycle time of Station B). This might be due to various factors like a slow worker, non-availability of parts or non-functioning machinery to name a few. It will put a lot of pressure on the workers in subsequent stations and has a negative impact on production, the most common being bottle necks [53].

Negative drift, meanwhile, occurs when the work assigned to a station is completed ahead of schedule and the station remains idle for the rest of the cycle time [54]. Although negative drift is seen also seen as a challenge to the production line, as it reduces line efficiency, it does not pose a big threat like positive drift which is considered to be costlier to the production line [5].

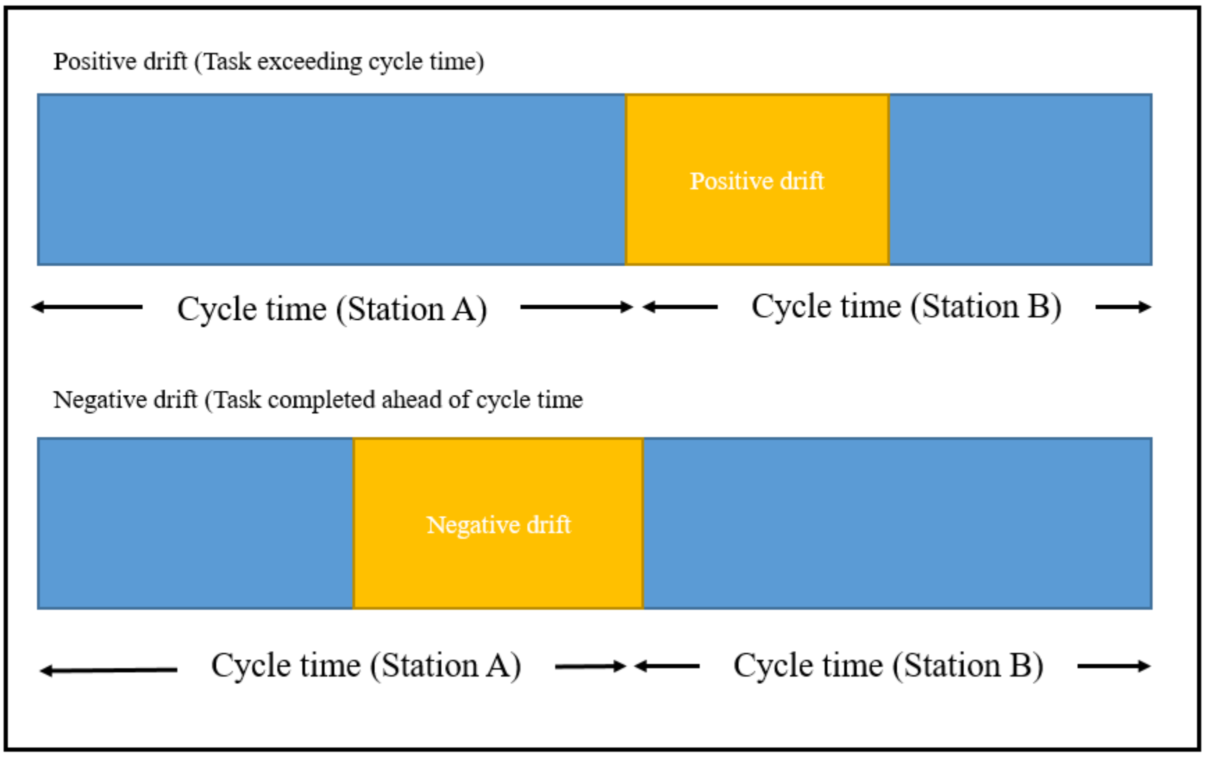


Figure 2. 4 Positive and Negative drift

One of the major goals of assembly line balancing is to ensure similar cycle time at each station, but manufacturing is undergoing a paradigm shift and moving towards a shift from mass production to mass customization and product variety.

In seeking variety in their products, consumers have gradually moved from consuming what is in stock to deciding what needs to be stocked [47]. This concept is often referred to as make-to-order systems [4].

2.4 Assembly Line Balancing Techniques

The previous section looked at assembly line balancing and the problems encountered in line balancing. In this section, the different techniques used for balancing assembly lines are analysed first. Secondly, a summary of research done in balancing assembly lines is done with respect to the classification by Sivasankaran and Shahabudeen [36].

2.4.1 Types of Assembly Line Balancing Techniques

Assembly line balancing techniques can be split into three general categories based on the type of problem that needs to be solved and the efficiency of the solution [20]. They are as follows;

- Optimal solutions – Optimal solutions are also defined as exact procedures. These are usually used for solving SALB problems. Several early researchers have contributed to finding optimal solutions for assembly line balancing problems [48]–[52]. The optimal solutions are based on mathematical models and are applicable to simple, deterministic and stochastic, S-type and U-type, assembly balancing problems.

Mixed integer programs, non-linear integer programs [53], goal programs, fuzzy goal programs [54] and constrained programs [55] are some of the optimal solutions used in conjunction with solvers like ILOG Cplex, LINGO, Xpress MP, MATLAB and GAMS

The effectiveness of optimal solutions is heavily reliant on the computation time required to solve a problem and this is where the main drawback of optimal solutions arise. They are not suited for large size problems [26] or real scale problems [28]. Therefore, it is seen that they are not very frequently used in multi model, deterministic or stochastic, assembly line balancing problems.

- Heuristics - Heuristics or approximate methods are developed to overcome the deficiencies of optimal solutions. They are not able guarantee an optimal solution, but achieve feasible solutions in acceptable computational time [56]. Some of the early vital contributions to heuristic approach was by Arcus [57], Dar-El [58], Helgeson and Brine [59], Hoffman [60] and Mansoor [61].

Single model S-type assembly lines with deterministic task times, were studied by Dar-El [38] and improved by Pannerselvam and Sankar [26] . The aim was to minimize the number of workstations for a given cycle time. Deterministic U-type assembly lines were researched on by Yegul et al [62].

Single model stochastic assembly lines pose a slightly sterner challenge as compared to single model deterministic assembly line. As a result, a few researchers have studied this area in depth. Gamberini et al[63] considered assembly rebalancing issues while Kottas and Lau [64] introduced the TOPSIS heuristic approach which was seen as an introduction to assembly lines being used for mass customization.

Due to the complicated design of all types of multi-mixed model assembly lines, heuristics are not often used in these balancing problems. The other drawback of the heuristic approach is that they are adapted to a specific problem and gets trapped in a local optimum, therefore fail to obtain a global optimum solution.

- Meta-Hueristics - The shortcomings of optimal solutions and heuristics are overcome using meta-heuristics. These procedures use optimal methods to find an initial solution and then use local search algorithms to create an improved solution. Some of the commonly used meta-heuristics are Genetic algorithms (GA), Tabu search algorithms (TS), Simulated Annealing (SA), Ant Colony Optimization. Shortest path algorithm and Bee algorithms

Single model deterministic straight type assembly lines were balanced with GA. Some of the early research contributions were by Rubinovitz and Levtin [65] and Kim et al [66]. An innovative GA with a local search algorithm was used by Gao et al [67] to minimize the cycle time of a simple assembly line balancing problem.

Single model stochastic assembly line balancing problems were solved using genetic algorithms by Tsujimura et al [68]. There has also been a use of fuzzy logic with GA to solve this problem. Gen et al [68] and Zacharia and Nearchou [69] have done research to this effect. Baykasoglu and Ozbakir [70] have shown how number of workstations can be minimized for a given cycle time for a U-type stochastic single model assembly line.

Multi/mixed model deterministic straight type assembly line problems have been solved using GA. Haq et al [71] and Sivansankaran and Shahabudeen [39] has done research stating this while multi/mixed deterministic U-type assembly lines problems have been solved by using GA by Kim et al [66]and Chutima and Olanviwatchai [72]. Research into the use GA to solve stochastic multi/mixed assembly lines was done by Xu and Xiao [53].

Tabu search algorithms are another method used for balancing assembly lines. Lapierre et al [73] has shown how TS algorithms can be used for SALBP-1 problems, while Erel et al [74]has shown evidence it being used for U-type single model assembly lines. Multi/mixed model assembly lines also use Tabu search algorithms. Ozcan et al [75] has done research showing how a two sided deterministic assembly line can be balanced using TS algorithms.

A summary of the research done on the different balancing problems and the balancing technique used is show in the next section.

2.4.2 Summary of ALB problems and solution methods

This section firstly tabulates the number of recorded studies done on the various assembly line problems discussed in Section 2.3.1 with respect to the balancing techniques discussed in Section 2.4.1. This is done in Table 1. Secondly, an analysis of the table is done to better understand the most common type of assembly line balancing problem and the preferred solution.

Table 2. 1 Number of studies done on the various assembly line problems with respect to the balancing techniques

	SMDS	SMDU	SMSS	SMSU	MMDS	MMDU	MMSU	MMSS
Mathematical modelling	7	2	3		4		1	
Heuristics	13	2	3	1		2		
Petri Net	2							
Genetic Algorithms	14		4	1	6	2	1	1
Simulated Annealing	5	1	2		2			1
Tabu Search	4			1	1			
Ant Colony Optimization	7	3			2		1	1
Shortest Path Algorithm	1	1	1					
TOTAL	**53**	**9**	**13**	**3**	**15**	**4**	**3**	**3**

A graphical representation of Table 1 shows that the Single Model Deterministic S- type assembly line problem is the most researched among the eight categories of assembly line balancing problems. This is depicted in Figure 2.5.

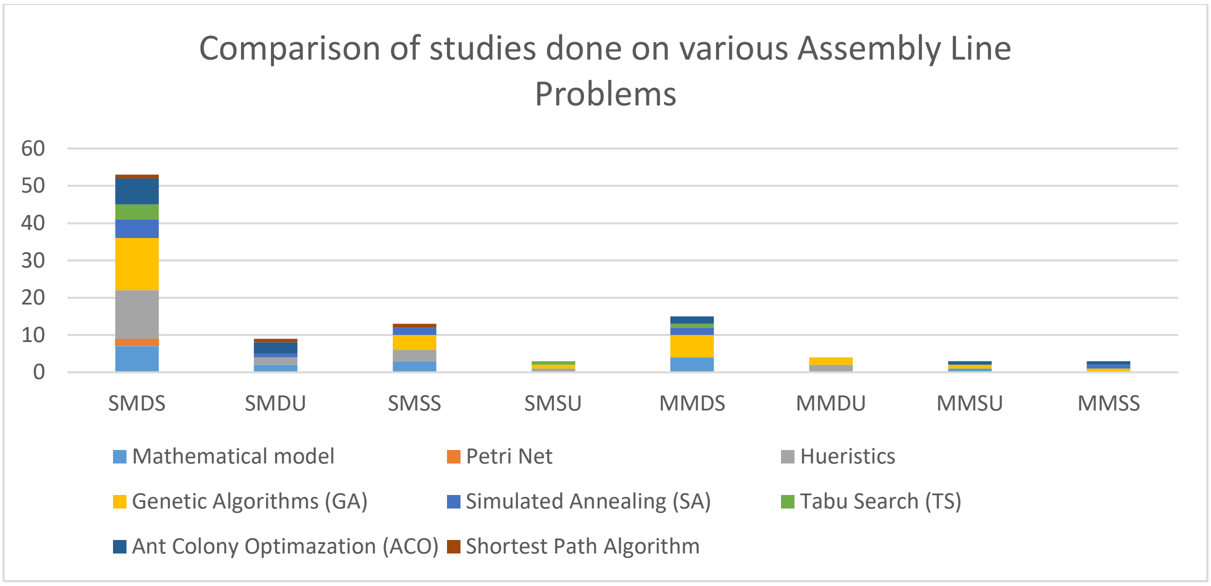


Figure 2. 5 Comparison of studies done on various assembly line problems

Figure 2.5 also shows that the Multi Model Stochastic S and U type assembly lines are the least researched along with the Single Model Stochastic U-type assembly lines. Another graphic shows that Meta heuristic methods like Genetic Algorithms are preferred over Heuristic and mathematical modelling techniques when it comes to solving assembly line balancing problems. This is graphically shown in Figure 2.6.

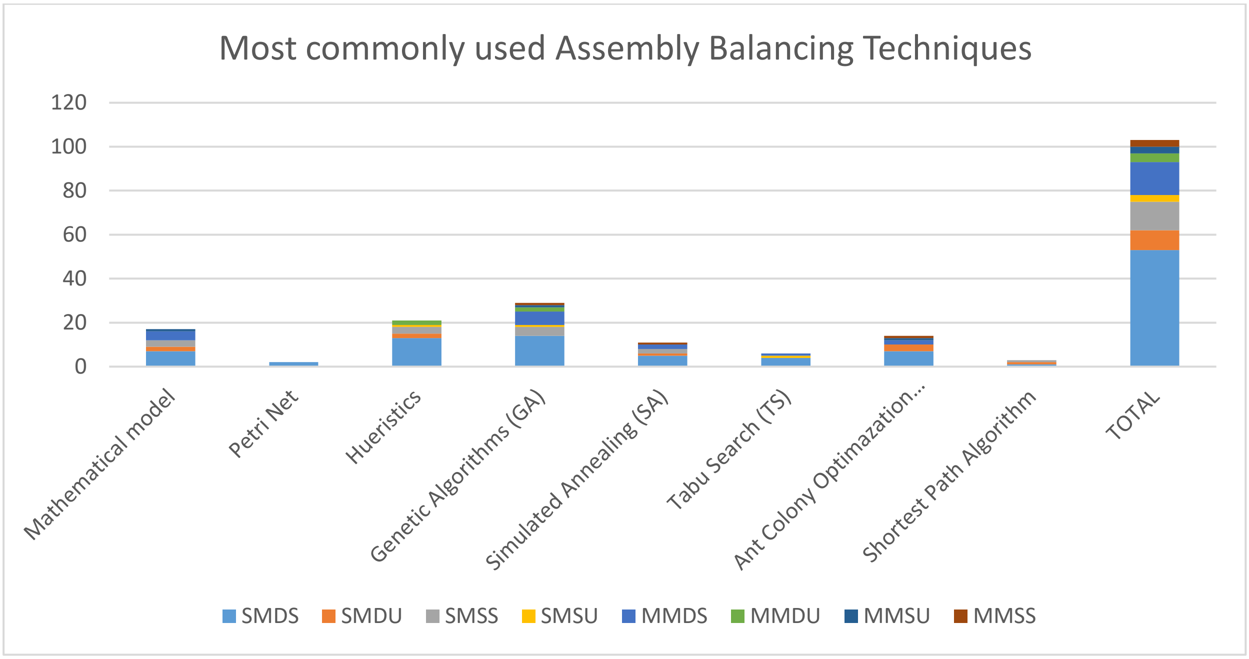


Figure 2. 6 Most commonly used assembly line balancing techniques

2.5 Limitations of Existing Research

This specific study focusses on optimization of real time multi mixed make-to-order assembly line to reduce positive drift. Make-to-order systems are by nature stochastic as their production cannot be pre planned and are dependent on the order by a client, hence they can only be manufactured by using a multi model stochastic (MMSS) assembly line.

MMSS assembly lines are split into two further categories being MMSS-1 type and MMSS-2 type problems. The two types are similar to the SALBP-1 & 2 type problems discussed in Section 2.3.1, where type 1 looks at problems concerned with reducing the number of workstations for a fixed cycle time and type 2 looks at problems concerned with reducing the cycle time for a fixed number of workstations.

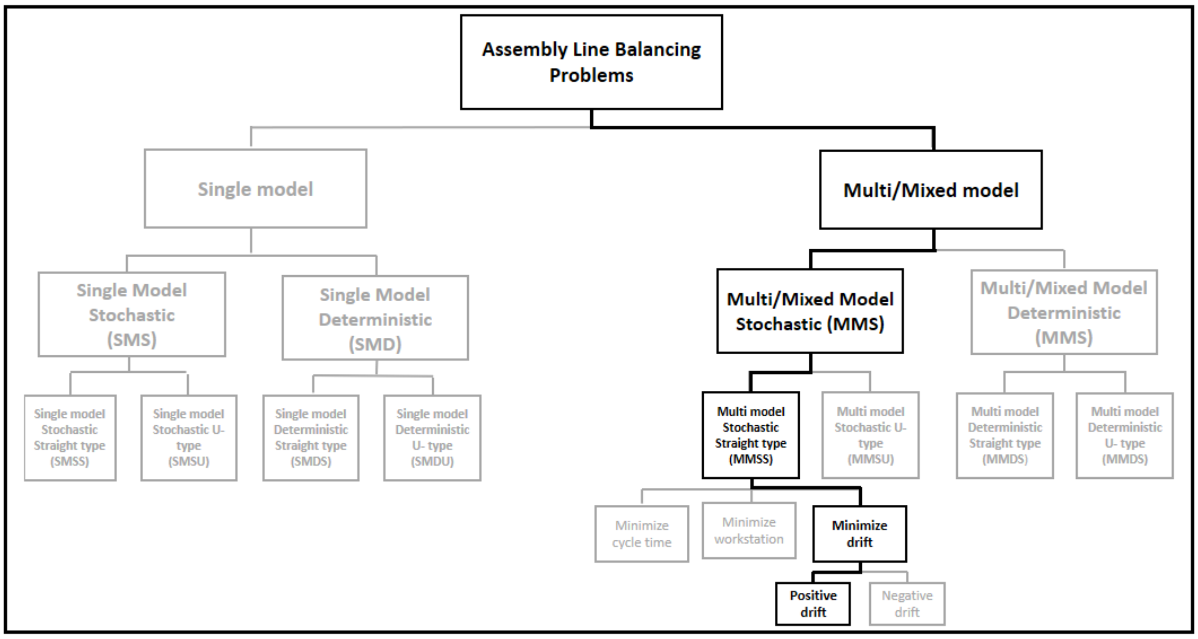
As seen from the summary in Section 2.4.2, at the time of writing this thesis, the amount of research done in Multi Model Stochastic Straight type assembly line problems was seemingly limited. The first of these studies is by McMullen and Taresewich [76]. They studied mixed model stochastic assembly line balancing problem with parallel workstations in which the objective is to design the assembly line such that the number of workstations or the cost is minimized for a given cycle time.

McMullen and Frazier [77] had earlier studied the balancing problem in which multiple product types scheduled in mixed-model fashion with stochastic task times and parallel workstations. The objectives here were the minimization of the total cost and maximization of the degree to which the desired cycle time is achieved [36].

Xu and Xiao [78] carried out a research in which an assembly line balancing problem with station lengths longer than the distance for which the conveyer moves within one cycle time is investigated in fuzzy environments, where operation times are assumed to be fuzzy variables. The objective is to minimize the positive drift time during the decision horizon.

Matanachai and Yano [79] also looked at the problem of positive drift on mixed model assembly lines in a multilevel production system. Their focus was mainly on the stability of the assembly line. Similar research by Tambe [80] included set-up time minimization and sequencing which was missing in the former study.

The major drawback of these studies is that there is very little research on combating positive drift and none when it comes to real time systems. This research aims to optimize a real time make-to-order multi mixed assembly line so as to reduce the positive drift. In doing so, there will, firstly, be addition to the existing knowledge on minimizing positive drift in MMSS assembly line systems. Secondly, there will be a new perspective on the impact of positive drift on real-time assembly line systems and how they can be optimized so as to negate the effects of positive drift. The research localization of this study is show in Figure 2.7.



Assembly Line Balancing Problems
Single model
Multi/Mixed model
Single Model Stochastic (SMS)
Single Model Deterministic (SMD)
Multi/Mixed Model Stochastic (MMS)
Multi/Mixed Model Deterministic (MMS)
Single model Stochastic Straight type (SMSS)
Single model Stochastic U-type (SMSU)
Single model Deterministic Straight type (SMDS)
Single model Deterministic U- type (SMDU)
Multi model Stochastic Straight type (MMSS)
Multi model Stochastic U-type (MMSU)
Multi model Deterministic Straight type (MMDS)
Multi model Deterministic U- type (MMDS)
Minimize cycle time
Minimize workstation
Minimize drift
Positive drift
Negative drift

Figure 2. 7 Research problem localization

Chapter 3 Research Methodology

3.1 Introduction

The aim of this chapter is to explain the research methodology used to address the challenges described in the limitations to the existing research described in Section 2.5. In this study, a make to order multi mixed model assembly line will be designed in Simulink and fed with real time inputs though a Cloud Interface. The model will then be initially run without optimization to measure the execution time. The model will then be optimized using an optimal method in MATLAB with respect to certain constraints.

In-order to design a multi mixed assembly line, a case study needs to be selected. In this study, a business plan to realise a water bottling plant at the Bloemfontein campus of the Central University of Technology [81], was chosen as the case study. This was done with two targets. Firstly, create a model which can serve as a platform for launching various studies into the research nice area of assembly line balancing and secondly, serve as a technical feasibility for the business plan. The next section elaborates on the primary considerations of the water bottling plant and why it was selected for this study.

3.2 The Water Bottling Plant- A case study

The business plan and the project schedule [82] for the water bottling plant was developed by Mr Fabian Maile, an exchange student from Hochschulle Aalen in Germany, under the supervision of Prof Herman Vermaak and Dr Nicolass Luwes. A conference paper detailing the business plan [83] was presented at an International conference.

The aim of the project was to setup a water bottling plant, hence forward referred to as the plant, in the Bloemfontein campus of the Central University of Technology so as to produce their own bottled water for use within the various faculties of the university, meetings, conferences and internal functions hosted by the university.

The water that was to be used by the plant was sourced from a bore well and was tested twice by independent bodies to ensure safety for human consumption. The details of the tests will not be discussed in this study as it falls outside the scope of this study. The plant, in discussion with various stakeholders, had to fulfil the following requirements;

- Able to bottle 5000 litres of water a month
- Contain all minerals
- Should be able to source and store water
- Should be able to produce 500ml and 750ml bottles
- Fill and cap the 500ml and 750ml bottles.

Based on these conditions a project and business plan was conducted and it was deemed that the plant would be economically viable for internal use initially. A road map was also setup to enhance the production of bottled water, so as to sell the bottles externally at a later stage. A three-dimensional visual model of the proposed plant was created and is shown in Figure 3.1

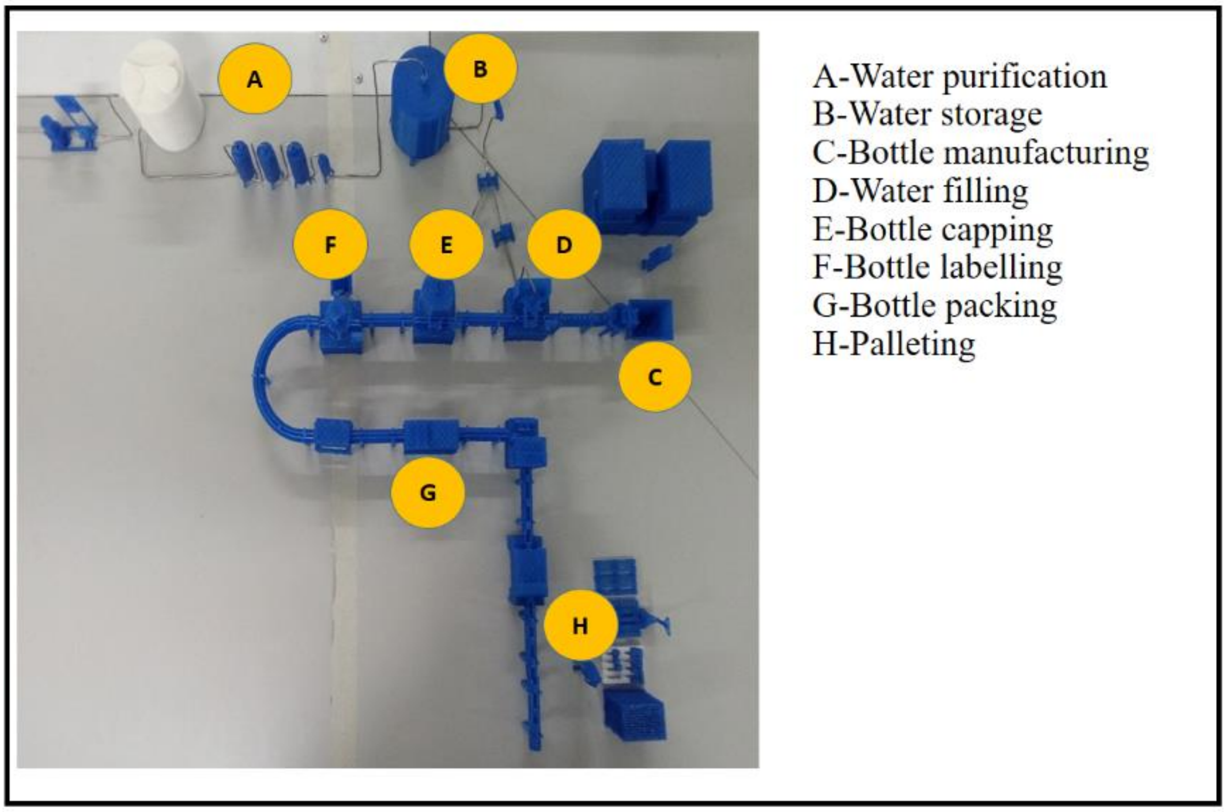


Figure 3. 1 3-dimensional scaled down printed model of the plant.

As mentioned previously. this study was initially setup to provide a technical feasibility for the business plan. The next section details how a primary model of the plant was created in Simulink as part of the technical feasibility study.

3.3 Primary Model Design of the Plant in Simulink

This section looks at how a primary model of the plant has been designed in Simulink. For the purposes of design, the 3D model shown in Figure 3.1 is split into three subsystems. This is shown in Figure 3.2.

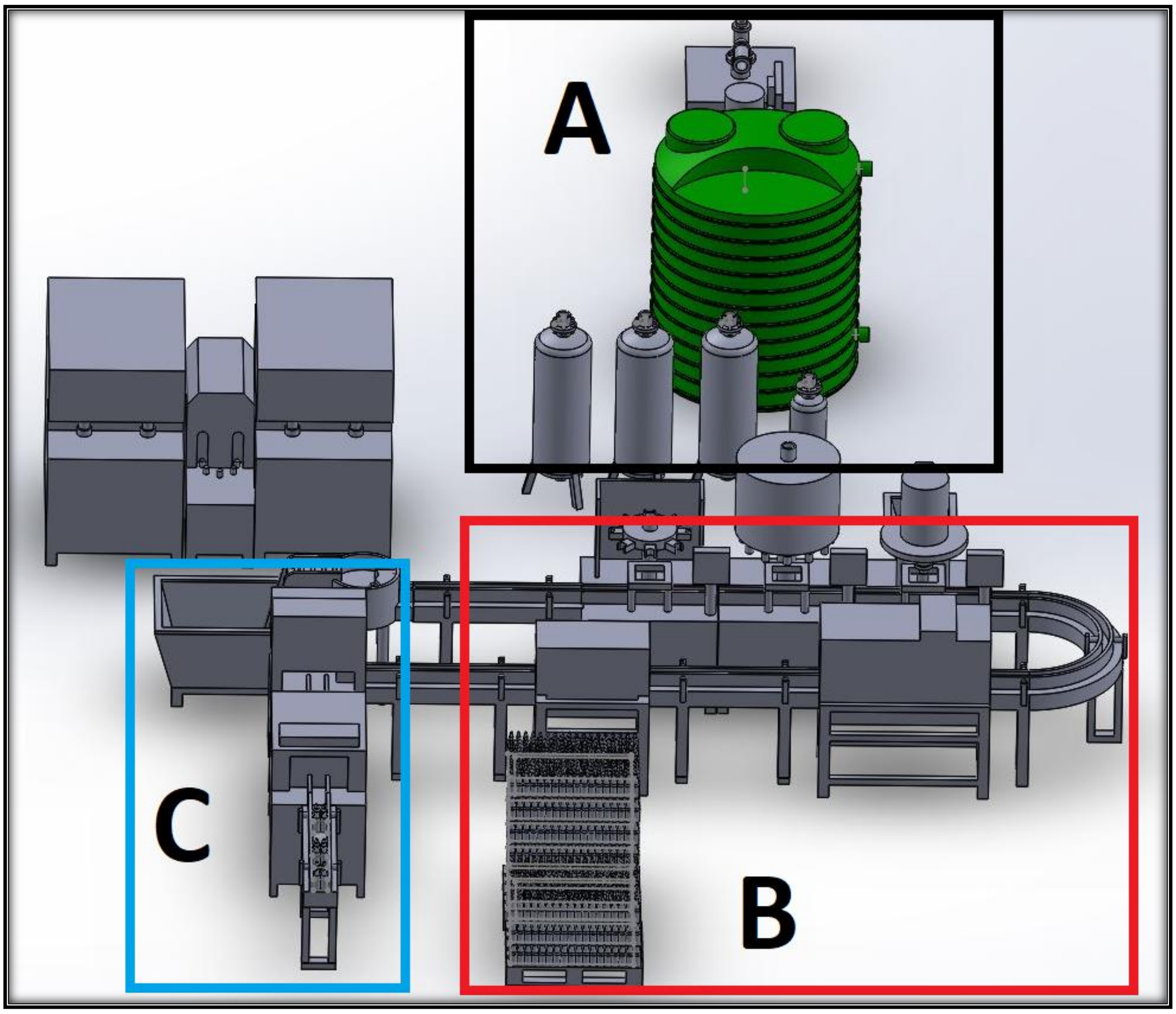


Figure 3. 2 3D model of the plant split into three subsystems

The three subsystems with their specific tasks are defined as follows;

- Subsystem A – Source and Storage tank
- Subsystem B – Bottle manufacturing and storage
- Subsystem C – Water filling

The Simulink plant model with the different subsystems is shown in Figure 3.3. It can be seen there that the three subsections described in Figure 3.2 are shown in the plant model.

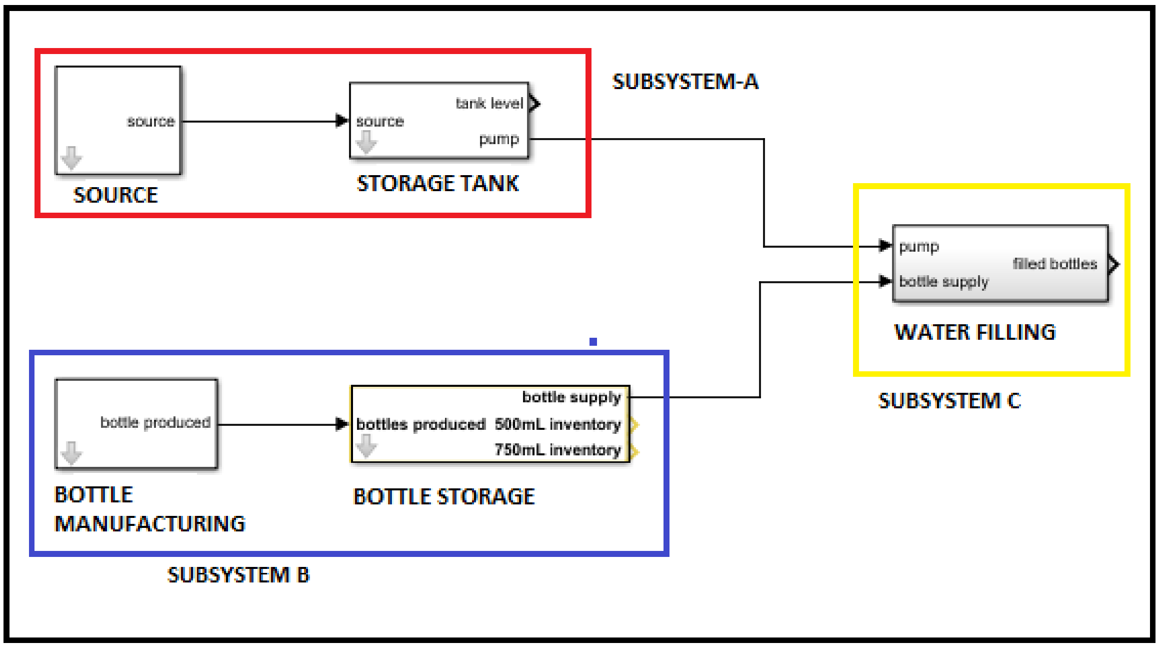


Figure 3. 3 Simulink model of the plant with three subsystems

Each element of the subsytem is described in the following section, starting with the subsystem A, followed by subsytem B and finally subsytem C. It is to be noted here that the plant function such as capping, labelling, packing and palletting which are described in the 3D model in Figure 3.1 have been ommitted from the Simulink model for convenience.

3.3.1 Source Subsystem

The source subsystem is designed to provide purified water to the plant. In comparison to the actual plant, the source subsystem will be analogous with unit A in Figure 3.1. For design purposes, the source subsystem is a masked subsystem, which means that parameters such as the flowrate and upper limit of water from the source system are user defined in the mask. A diagram representing the different blocks which make up the source subsystem is given in Figure 3.4.

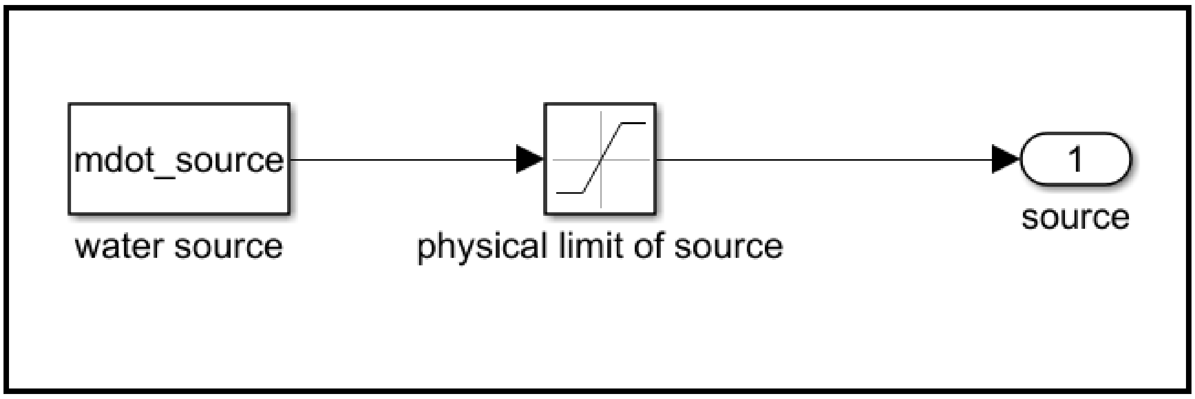
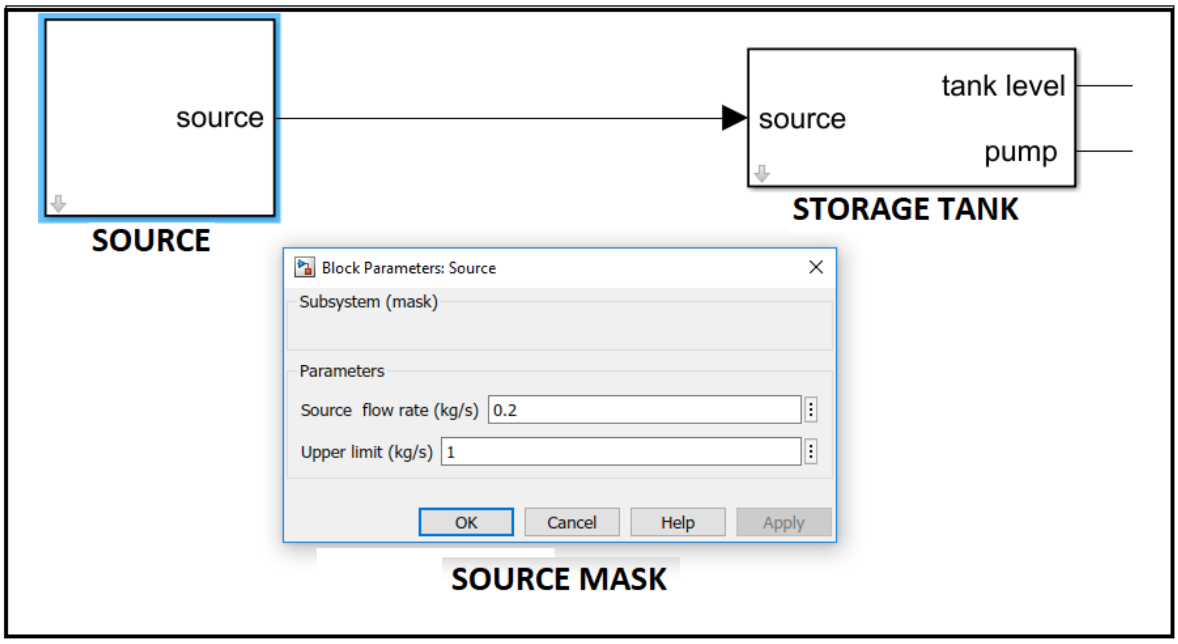


Figure 3. 4 Source subsystem

As seen from the Figure 3.4, the source subsystem consists of three blocks, which are a constant block, a saturation block and an outport block.

The constant, which is defined as *mdot_source*, acts as the physical source of water. The flow rate of the source is measured in kilolitres per second and is output to a saturation block. The saturation block will have an upper limit and a lower limit. As a result, when the flow rate from the water source exceeds the water source, it gets clipped to the upper limit of the saturation block. The lower limit of the saturation block is set to zero. The output from the saturation block is fed to the Outport which will be the input to the storage tank subsystem. Figure 3.5 shows the Source mask with user defined values of flow rate and upper limit.



Source flow rate (kg/s) 0.2

Upper limit (kg/s) 1

Figure 3. 5 Source subsystem mask with user defined values

3.3.2 Storage Tank Subsystem

The storage tank subsystem needs to take purified water from the source subsystem and pump it into the water filling subsystem. These conditions make it a continuous state subsystem. Continuous states, in the context of Simulink, refers to a variable whose value is determined through numerical integration of its derivative with respect to time. In this specific case the water coming from the source into the tank needs to be deducted from the water flowing out of the tank through the pump to obtain the net water flow.

This is mathematically described in Equation 3.1. The net water flow needs to be integrated to obtain the volume of water which is retained in the tank. This is mathematically described in the Equation 3.2.

$$m\_n\dot{e}t(t) = m\_sou\dot{r}ce(t) - m\_pu\dot{m}p(t) \qquad (3.1)$$

$$m\_\dot{n}et(t) = m\_net(t) + u(t) \qquad (3.2)$$

Where;

$m\_n\dot{e}t(t)$ = Net water stored in tank

$m\_sou\dot{r}ce(t)$ = Water coming from the source

$m\_pu\dot{m}p(t)$ = Water going out of pump

The tank level is calculated as a percentage using the following formula and represented as Equation 3.3

$$Tank\ level = \left[\frac{m\_net(t)}{Vmax\ X\ 1000}\right] X\ 100 \qquad (3.3)$$

The expanded block diagram of the storage tank subsystem in shown in Figure 3.6

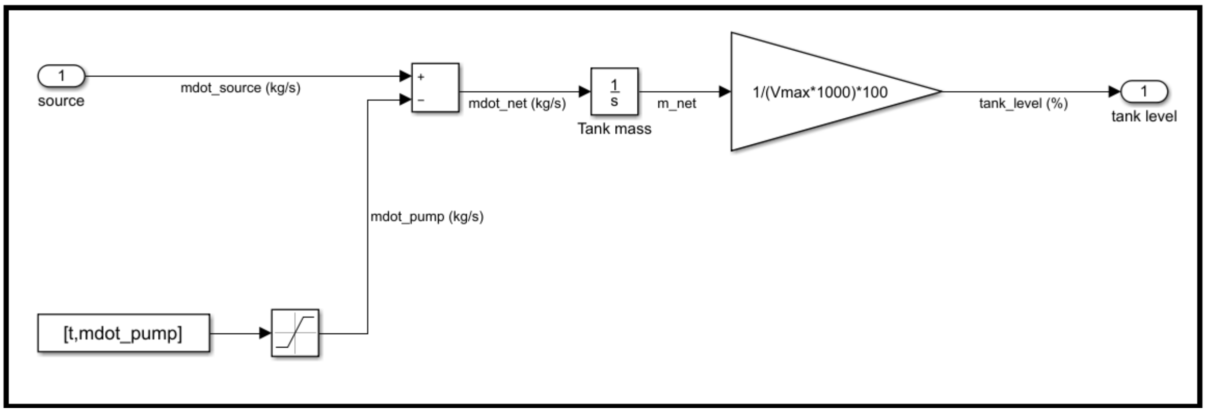
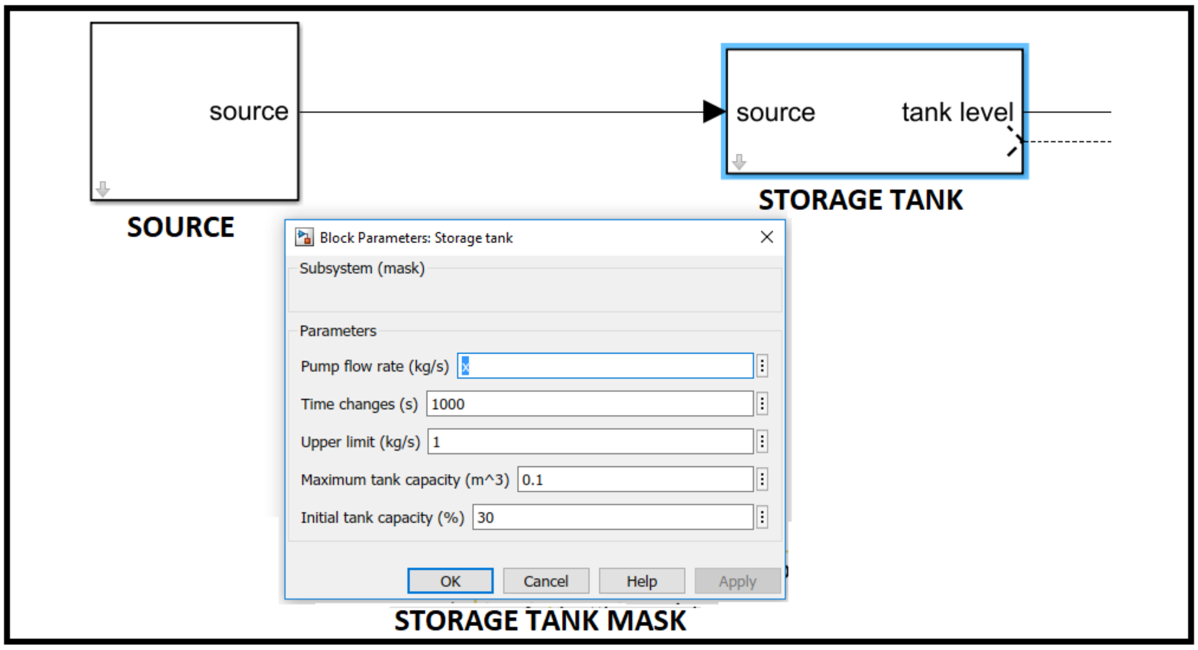


Figure 3. 6 Storage tank subsystem

As it was the case in the Source subsystem, the storage tank subsystem is also a masked subsystem, which allows either the user to enter values or read values from an external MATLAB script. The storage tank mask is shown in Figure 3.7.



Pump flow rate (kg/s) x
Time changes (s) 1000
Upper limit (kg/s) 1
Maximum tank capacity (m^3) 0.1
Initial tank capacity (%) 30

Figure 3. 7 Storage tank subsystem mask with user defined values

3.3.3 Bottle Manufacturing Subsystem

This subsystem along with bottle storage subsystem are independent of the source and storage tank subsystem. The design parameters require that bottles be manufactured in two sizes, 500ml and 750ml. For design purposes, the bottles were initially manufactured in batches of six. The block diagram representation of the bottle manufacturing subsystem is shown in Figure 3.8.

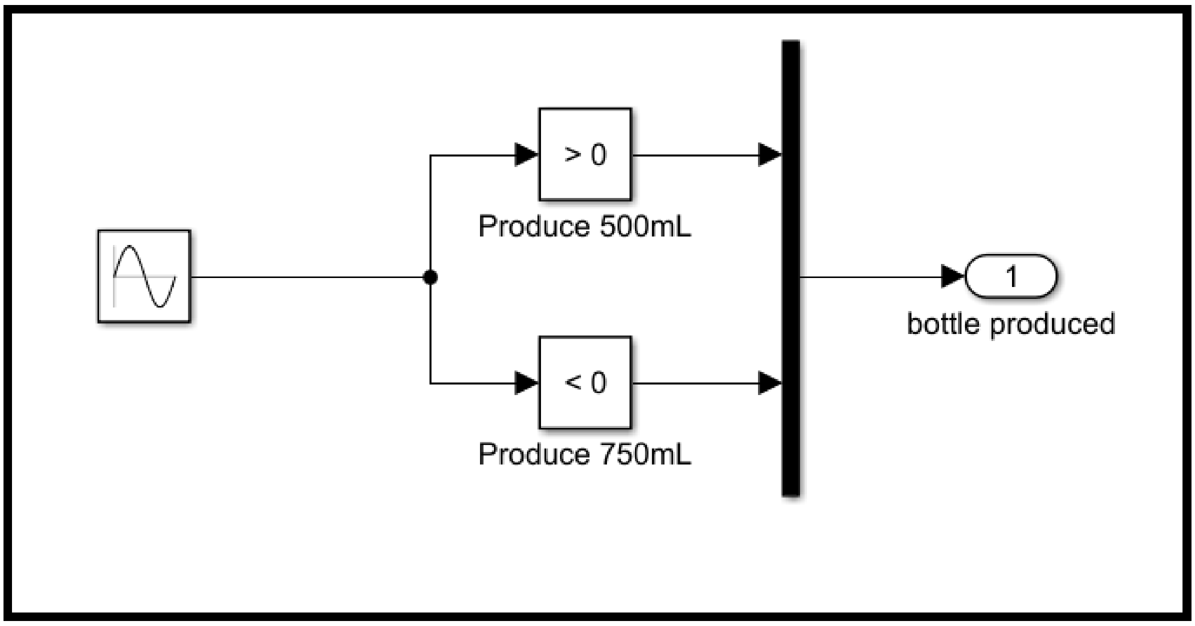
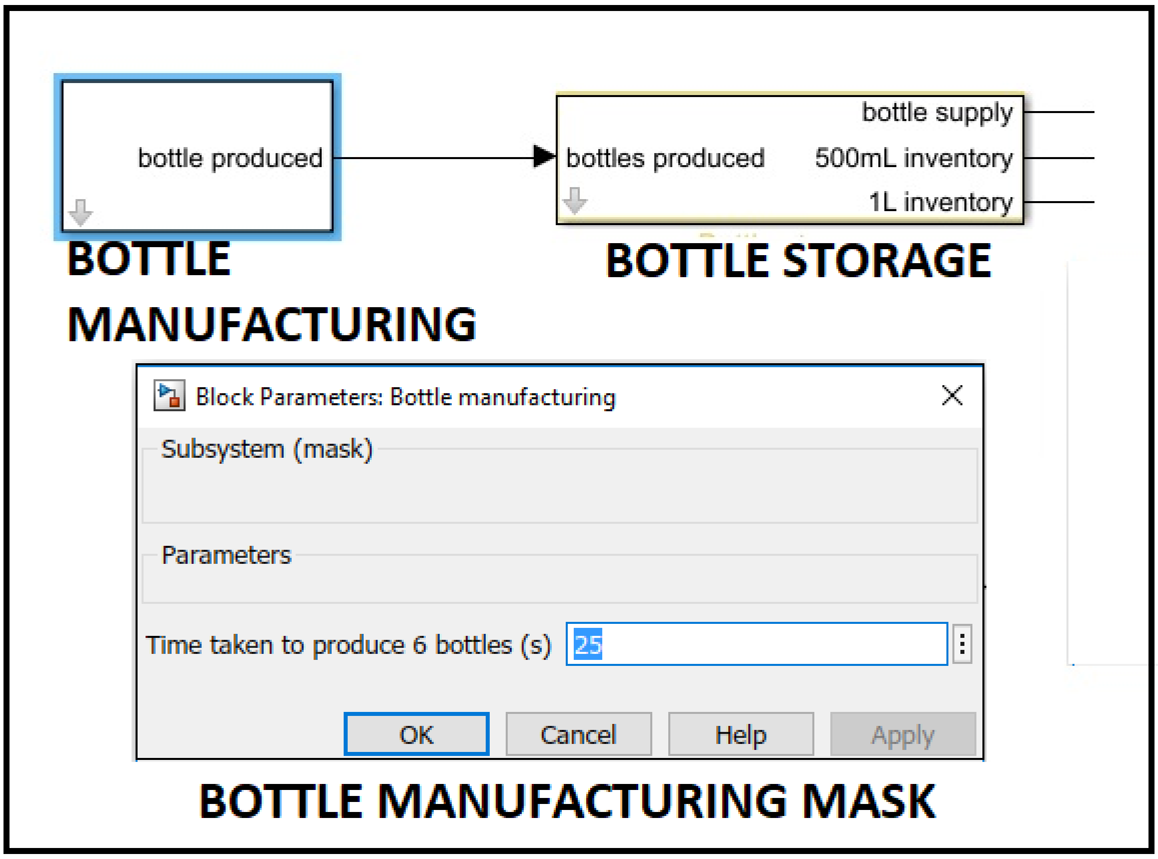


Figure 3. 8 Bottle manufacturing subsystem

In order to meet the design requirement, the following modelling logic is used. A sine wave is passed through a compare with zero block with an output HIGH Boolean, where the rising edge is considered greater than zero and the falling edge less than zero.

During the positive half cycle of the sine wave, the output connected to 500ml inventory produces a HIGH for the time set in the mask, while during the negative half cycle of the sinewave, the output connected to the 750ml inventory remains HIGH for the time set in the mask. The two outputs are connected to a multiplexer and provided to the bottle storage subsystem.

The time taken to manufacture the six bottles is defined under a user defined mask which is shown in Figure 3.9



Block Parameters: Bottle manufacturing

Subsystem (mask)

Parameters

Time taken to produce 6 bottles (s) 25

Figure 3. 9 Bottle manufacturing subsystem mask with user defined values

3.3.4 Bottle Storage Subsystem

The output of the bottle manufacturing subsystem is fed to the bottle storage subsystem. As explained in the design considerations, the bottle storage needs three input components for each type of bottles. The first and primary input is that of the bottle manufacturing unit. The second input is the number of bottles that have been filled with water already and finally the initial count of bottles. The initial number of bottles will be user defined for modelling purposes, hence defined under the subsystem mask.

The output of the bottle manufacturing subsystem is provided to a de-multiplexer, which splits it into the 500ml and 750ml bottle types. These are connected to a 500ml and 750ml inventory. The inventories are designed using a triggered subsystem. The triggered subsystem has a constant output of six, which relates to the batch the bottles are made in the bottle manufacturing subsystem. This output is added to initial number of bottles every time the system is triggered on the rising edge.

The bottle storage unit has so far got the initial input as well as the input from the bottle manufacturing subsystem. The missing input is the number of bottles used. In order to achieve this, an output is obtained from a GoTo block defined in the bottle filling subsystem. The output from the GoTo block is fed to a triggered subsystem, which removes a bottle every time it is pulsed. This output is fed to an adder with the initial inventory and the bottles produced. Figure 3.10 depicts the Simulink blocks used in defining this subsystem.

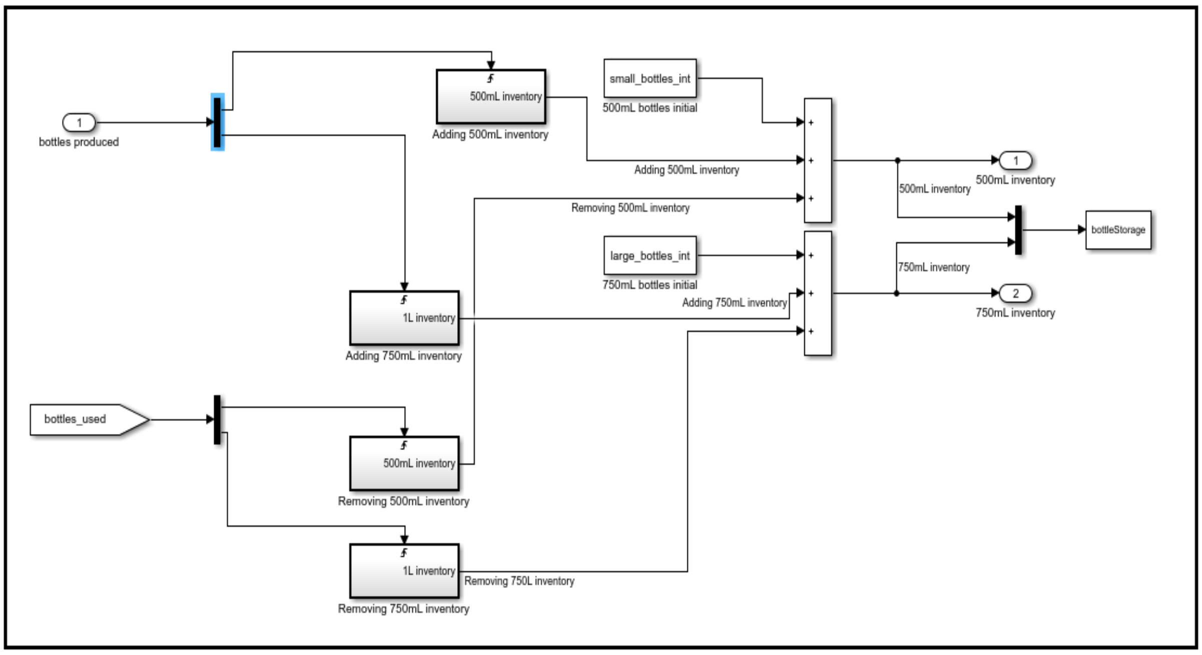
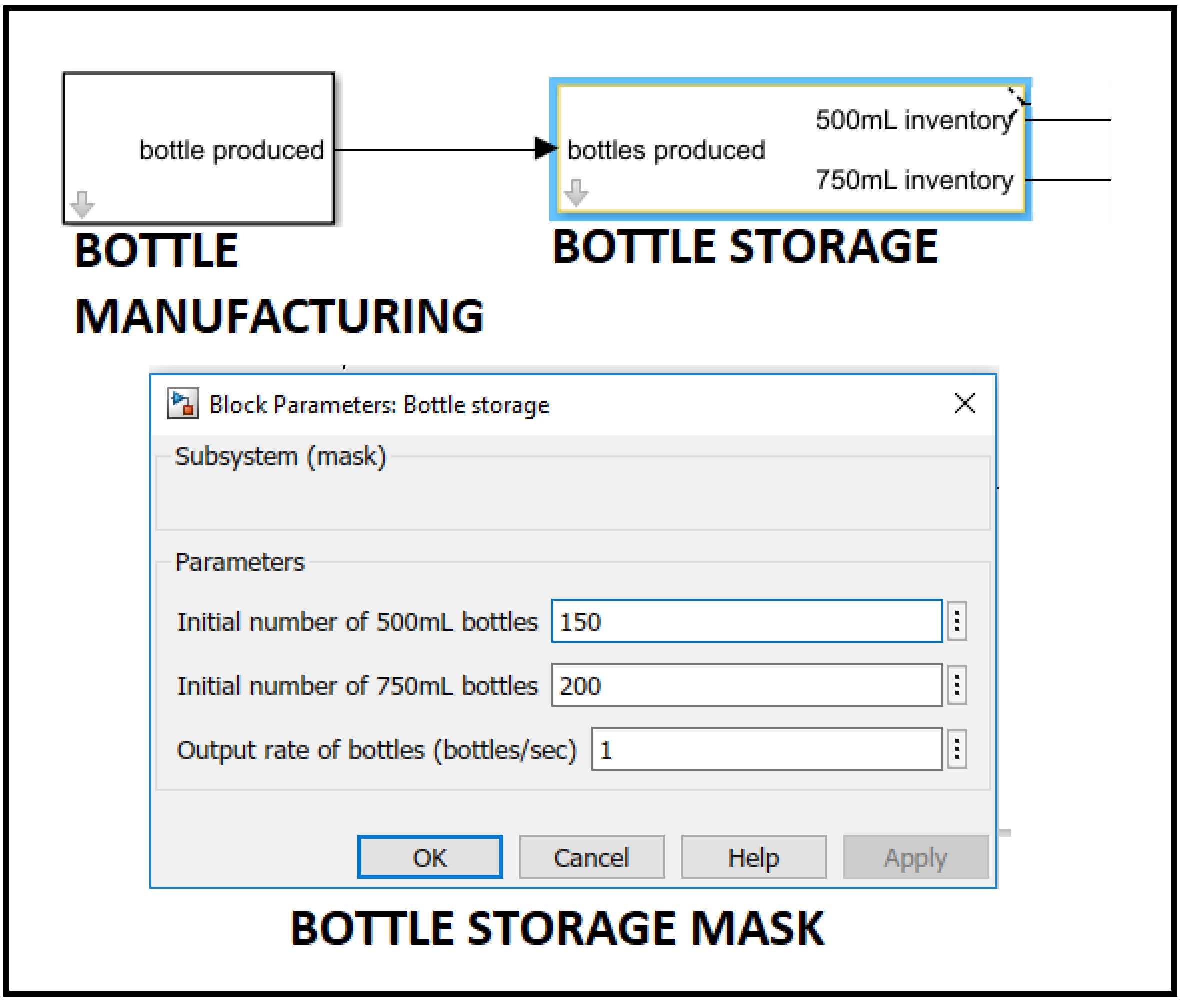


Figure 3. 10 Bottle storage subsystem

As in the other subsystems, the bottle storage subsystem also has a user defined mask in which the initial number of 500ml and 750ml bottles can be defined along with the output rate of bottle production. This is depicted in Figure 3.11.



Initial number of 500mL bottles: 150
Initial number of 750mL bottles: 200
Output rate of bottles (bottles/sec): 1

Figure 3. 11 Bottle manufacturing subsystem mask with user defined values

3.3.5 Water Filling Subsystem

The input of the water filling subsystem is the pump from the storage tank subsystem. The output of the pump, which is a continuous system, needs to be integrated to obtain the volume of water. This is then provided to a 500ml or 750ml bottle. The successful operation of this subsystem is dependent on how the model is able to distinguish between 500 and 750ml bottles.

In order to distinguish between the two, a switch block is used. The switch block is connected to constant blocks which distinguish between the 500ml and 750ml bottles. The threshold of the switch block is kept at 5 and connected to triggered subsystem which acts as a modulus 12 counter. The switch outputs 500ml bottles for counts from 0 to 5 (6 bottles) and outputs 750ml bottles from counts 6-11 (6 bottles). Once 12 bottles are reached, the subsystem is triggered therefore returns back to zero and restarts the process.

As the two different bottles get filled, the number of bottles used must be counted. This output is fed back to the bottle storage subsystem through a GoTo block as explained in the bottle storage subsystem. The count of bottles used is maintained using a similar logic to that used in filling the bottles, the only difference being that two switch blocks are used, one each for filling the 500ml and 750ml bottle. The water filling subsystem is depicted in Figure 3.12.

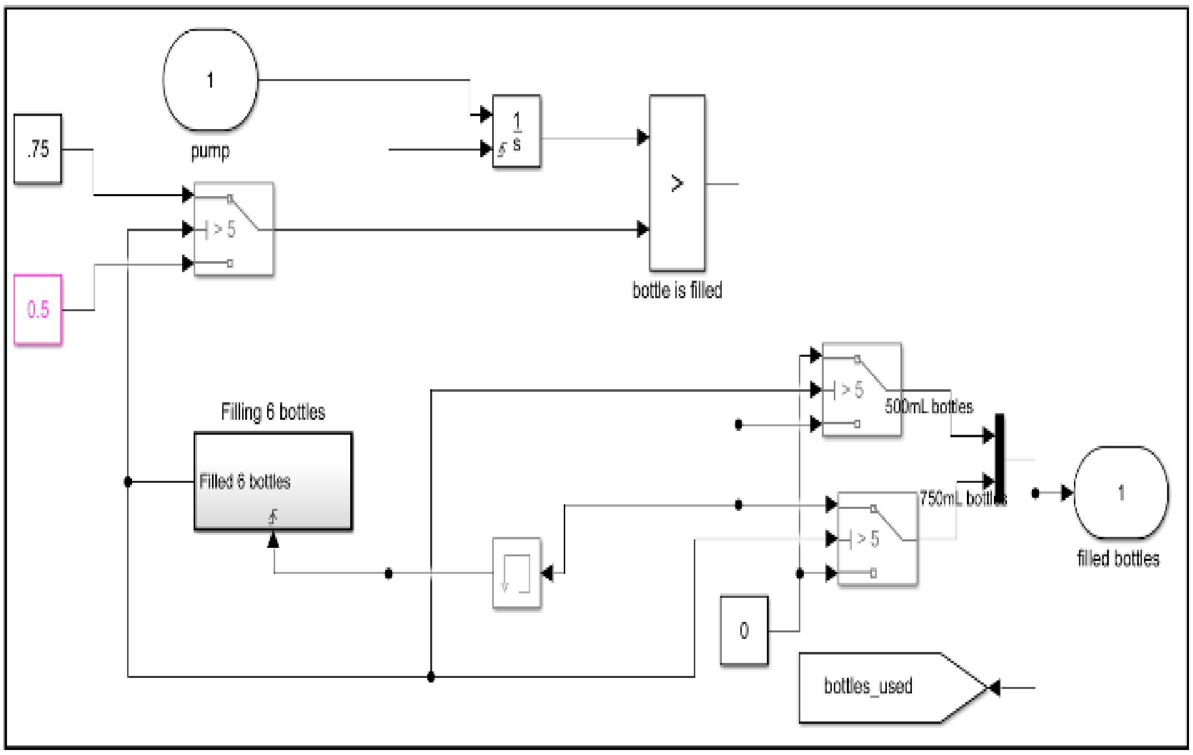


Figure 3. 12 Water filling subsystem

The exact results of running the model will be shown in the next section, but the following conclusions could be read with the intial model.

- The model could fill 500ml and 750ml bottles in different cycles without overlapping
- The number of bottles used for filling and that remains in the storage never went below zero. If the latter was to occur, the process would fail in real time.
- The model was able to distinguish between 500ml and 750ml bottles and produce them in batches of six.

However, the following drawbacks were noted for the model;

- The results were for a specific set of user defined mask values, there would be instances where the number of bottles remaining in the inventory would go below zero.
- The bottles were filled in batches of six, this would not allow for customization.

- The filling of bottles was not optimized.

As a result, the next modelling challenge will be to customize the plant model, so that it can fill bottled water according to customer orders. This setup is discussed in the next section. A paper detailing the basic model of the plant [84] was submitted, accepted and awaiting publication at the time of writing this thesis.

3.4 Customising the plant model in Simulink

The previous section showed how a mixed model assembly line in the form of a water bottling plant could be modelled in Simulink. The drawback of the model is that it did not have customized or make-to-order inputs. This section looks at the changes that were initiated to the primary model in order to make the customized inputs.

As mentioned previously, in the primary model, the bottles were filled in batches of six using a modulo 12 counter to switch between 500ml and 750ml bottles. This is depicted in Figure 3.12. In order to customize the inputs, a MATLAB script is first written to record the customer requirements in tabular format. The MATLAB script can be called into the model to serve as the input.

In order to read the table into the model, the modulo 12 counter needs to be replaced with a subsystem that can read the values in customer requirements table to the new model. The subsystem referred to as 'Customer requirements' is shown in Figure 3.13.

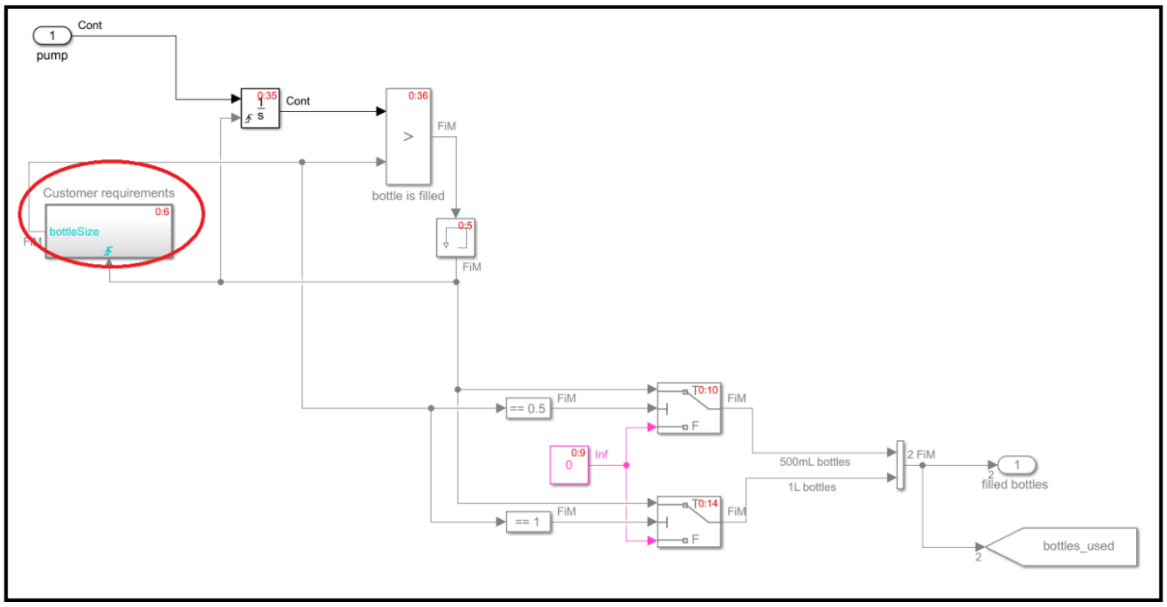


Figure 3. 13 Water filling subsystem with customer requirements subsystem

This can be achieved by using a one-dimensional lookup table. The data from the customer requirement table can be transposed and flattened to appear as a row of information to the table. A relational table with a memory block can be used to check if the number of bottles in each row has been reached.

As soon as the number of bottles in the first row has been achieved, a trigger element can be set up to firstly index the data and move to the next row in the look up table. This can be continued till the last row in the lookup table has been read into the model and defined in the index. The distinction between 500ml and 750ml bottles can be made by calling a function in the model which checks the index where the data has been read from. By default, the 500ml bottles will be indexed in the odd rows while the 750ml will be indexed in the even rows. A diagrammatic representation of the customer requirements subsystem is shown in Figure 3.14

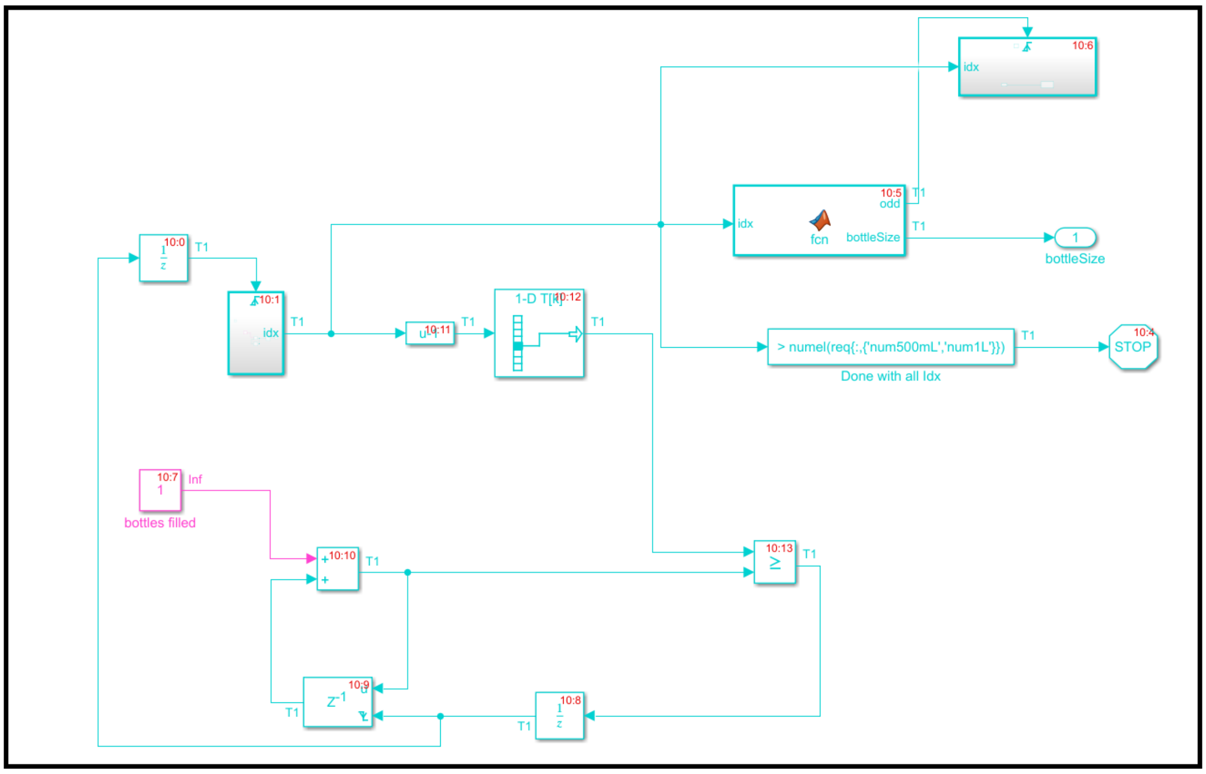


Figure 3. 14 Customer requirements subsystem

After the customer requirements have been customized, the output of the filled bottles from the subsystem in shown Figure 3.14 will be provided to filled bottle storage subsystem. The filled bottle storage system checks to ensure that the customer requirements have been successfully met. This is represented in Figure 3.15. A paper detailing the customized model [85] was published.

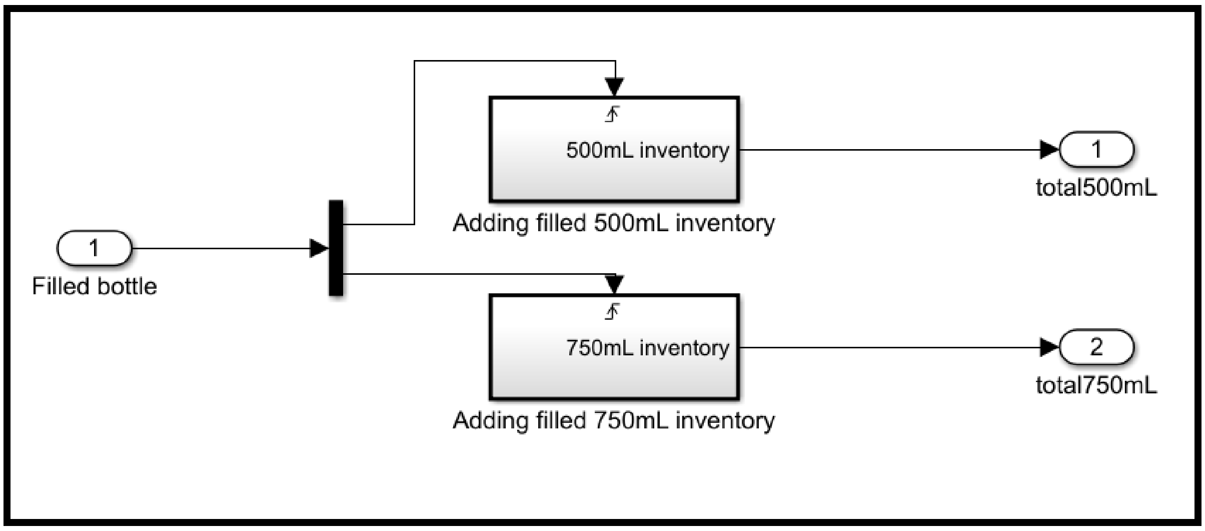


Figure 3. 15 Filled bottle storage subsystem

3.5 Designing a Make-To-Order System using a Cloud Server

The input customization defined in Section 3.4 is an important addition to the model as it makes the model flexible. However, the challenge still persists that inputs are not make-to-order. A make-to-order system, discussed in Section 1.1, is one which gives customers the ability to make their choice of the inputs.

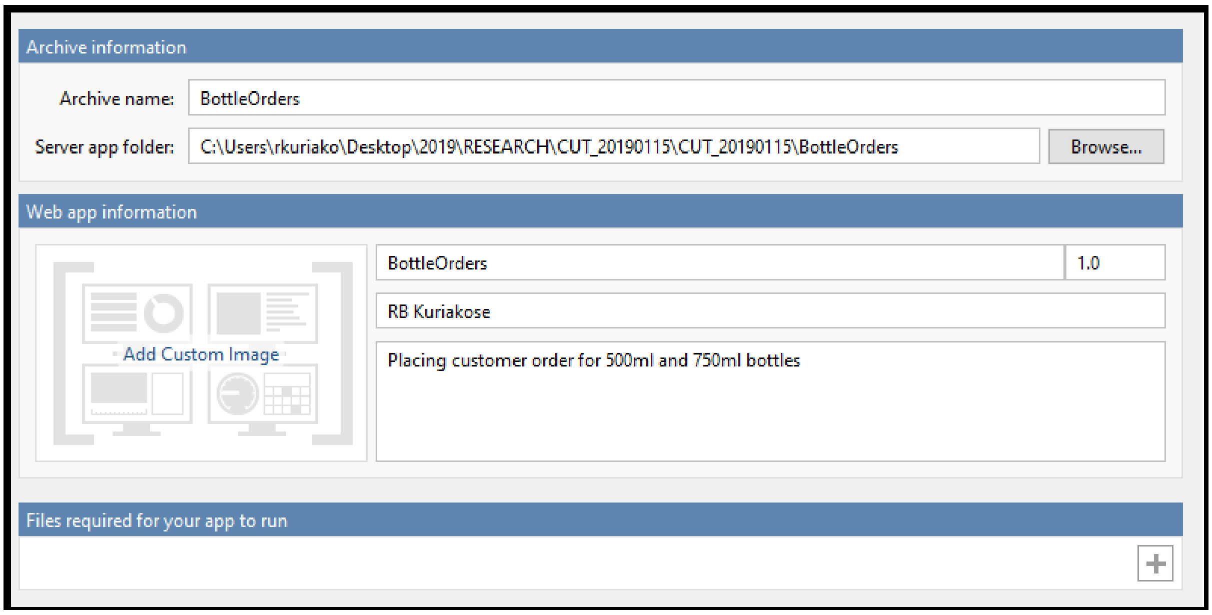
In this specific model, customers need to be given the option of choosing the number of 500ml or 750ml bottles that they require. Section 3.4 described how the primary model was altered to create customer requirements into a tabular format. This Section looks at how a web application (app) is developed and hosted on a cloud server.

Web apps are created within the MATLAB environment as applications which can be run from any web browser as a Graphic User Interface (GUI) which can be accessed using a secure Uniform Resource Locator (URL). The GUI contains information such as the name of the client, the number of 500ml and 750ml bottled water required by the customer and the required delivery date.

The data captured by the GUI is send and stored on the Cloud server. The stored data is then passed into the customer requirements subsystem discussed in Section 3.4 and depicted in Figure 3.13. This part of the thesis firstly discusses how the web app was developed and deployed on the server. Then it looks at how the data pertaining to the customer requirements was transferred to the model from the server.

3.5.1 Developing and deploying a MATLAB Web App

The MATLAB web app is developed using the MATLAB compiler and hosted using the Web App Server. As mentioned briefly in Section 3.5, web apps are MATLAB apps that can run in a web browser. They are developed using the App compiler function in the MATLAB. The App compiler allows the user to define the server app folder. The compiler is shown in Figure 3.16.


Archive information

Archive name: BottleOrders

Server app folder: C:\Users\rkuriako\Desktop\2019\RESEARCH\CUT_20190115\CUT_20190115\BottleOrders

Web app information

BottleOrders 1.0

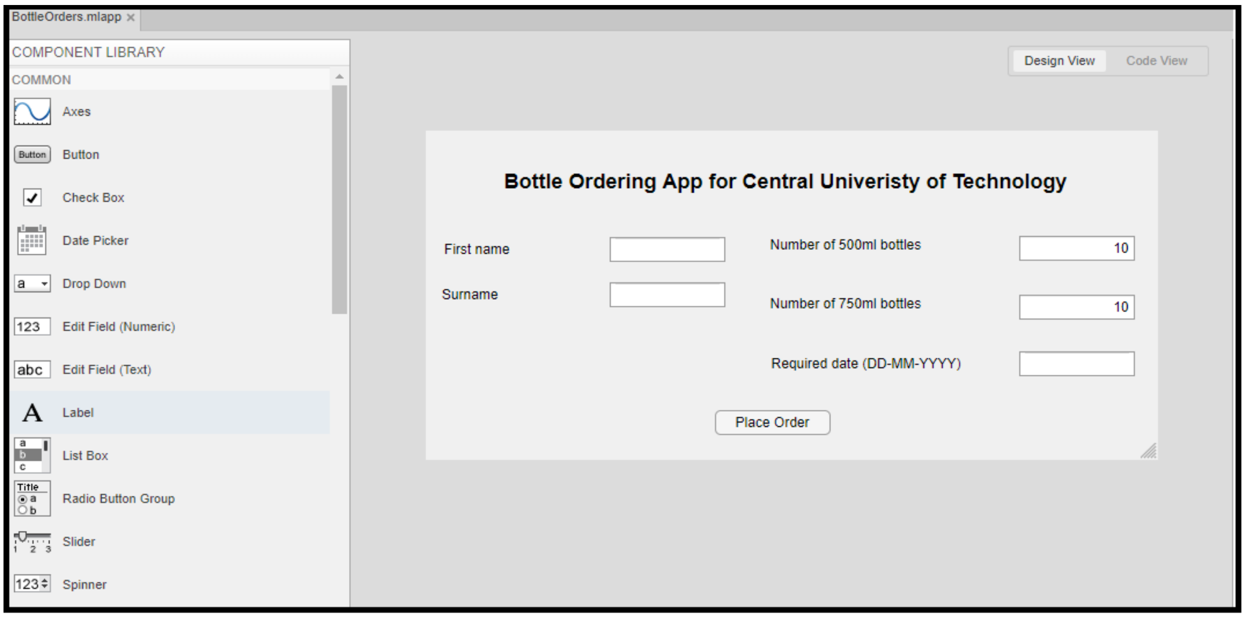
RB Kuriakose

Placing customer order for 500ml and 750ml bottles

Files required for your app to run

Figure 3. 16 Creating a web app using the compiler

The web app can now be opened in the app designer. In the app designer, the different blocks that are needed to define the app can be set out using drag and drop options from the component library. The web app created for ordering bottles is shown in Figure 3.17. As seen from Figure 3.17, the web app contains blocks where the customer can enter their name, number of 500ml and 750ml bottles required and the required date of delivery. On completion they can place the order.



Bottle Ordering App for Central Univeristy of Technology

First name		Number of 500ml bottles	10
Surname		Number of 750ml bottles	10
		Required date (DD-MM-YYYY)	

Place Order

Figure 3. 17 Web app for ordering bottles

3.6 Optimizing the plant model in MATLAB

The previous parts of the thesis showcased how the plant model was designed, firstly, to be mixed model and secondly to be make-to-order or customised. This section focuses on the optimization of the model. A broader study into assembly line balancing was done in Section 2.4 which showed that there was very little research into Multi Model Stochastic S-type (MMSS) assembly lines.

This part takes cognizance of the apparent disparity in the research and aims to fill the gap by initially examining what is optimization and its different types. It then focusses on Real Time Optimization (RTO). Thirdly, the developed model and the constraints governing it are stipulated. Finally, an apt modelling technique in MATLAB is chosen to address the optimization problem at hand.

3.6.1 Introduction to Optimization problems

Optimization is defined [86] as the process of maximizing or minimizing an objective function with or without constraints with respect to certain input variables. The basic components of an optimization problem [87] are the following;

- An objective function – This is a function which expresses, in mathematical or non-mathematical form, the model which needs to be minimized or maximized. A model can have no objective functions or multiple objective functions. The event that there are no objective functions, is often referred to as a feasibility problem while multiple objectives are often formulated as single equation with weighted combinations of the objectives.

- Variables – These are a set of values which control the value of the objective function. Like objective function, a model can have single or multiple variables. However, variables are an essential component as without variables, the objective functions and constraints cannot be defined.
- Constraints – Also referred to as boundaries or limitations that have to be adhered to when minimizing or maximizing an objective function. Constraints can be equality or non-equality.

A standard optimization problem [88][89] can be defined as follows;

Find,

$$x = x^1, x^2, \dots \dots , x^n$$

Which minimizes or maximizes,

$$f(x)$$

Subject to constraints,

$$g_j\,(x) \leq 0$$

For $j = 1, \dots .., m, and$

$$l_j(x) = 0$$

For $j = 1, \dots .. p.$

Here,

$x = Design\ vector$

$f(x) = Objective\ function$

$g_j\,(x) = Inequality\ constraints$

$l_j(x) = Equality\ constraints$

$n = number\ of\ variables$

$p = m = number\ of\ constraints$

If in an optimization problem $p + m = 0$, then the problem is referred [90] to as an unconstrained optimization problem.

3.6.2 Classification of Optimization problems

Optimization problems can be classified according [91] to the following parameters;

- Type of constraints
- Nature of variables
- Physical structure of the problem
- Nature of equations involved in defining the problem
- Number of objective

3.6.2.1 Classification based on type of constraints

According to the number of constraints in an optimization problem, the categorization is of two types being;

- Constrained optimization problems – optimization problems in which there are more than one constraints.
- Unconstrained optimization problems – optimization problems where there are no constraints.

3.6.2.2 Classification based on nature of design variables

There are two broad categories in this classification. They are listed as follows;

- Static optimization problem – in this category, the aim is to determine a set of design parameters that make the objective function to be minimum or maximum subject to certain constraints.
- Dynamic optimization problem – in this type of an optimization problem, the aim is to determine a set of design parameters, which are all continuous functions of other parameters, that minimizes an objective function subject to a set of constraints.

3.6.2.3 Classification based on physical structure of problem

Based on physical structure, optimization problems can be classified into the following categories;

- Optimal control problems – is an optimization problem in which there are a number of stages. The optimization of each stage depends on what happened in the preceding stage in a prescribed manner.
- Non-optimal control problems – in this type of an optimization problem, there are stages in the optimization. The total objective function is made into a single equation and minimized or maximized subject to a set of constraints.

3.6.2.4 Classification based on the nature of equations involved

Based on the nature of equations the optimization problems can be split into at least four categories. They are defined as follows;

- Linear programming problem (LP- problem) – Here, the objective function and all the constraints are linear functions of the design variables.
- Nonlinear programming problems (NLP- problem) – In this type of an optimization problem, the functions among the objectives and constraint functions exhibit a nonlinear relationship.
- Geometric programming problems (GMP- problem) – is one in which the objective function and constraints are expressed as polynomials in ‘x’.
- Quadratic programming problems – is classified as a nonlinear problem with quadratic objective functions and linear constraints.

3.6.2.5 Classification based on the number of objective functions

Based on the number of objective functions, optimization problems can be classified as single or multi objective programming problems.

3.6.3 Real Time Optimization (RTO)

The model that has been designed this far in this study has customized inputs. In Section 3.6, it will be detailed how these inputs will be sourced from a Cloud server. This would mean that model will have real inputs. Therefore, this section of the study focusses on Real Time Optimization (RTO) which is a part of the process control and optimization in manufacturing.

RTO has the ability to integrate [92] process measurements into an optimization framework, which has been considered as the biggest challenge to process control and optimization. By means of introducing RTO, process optimization no longer relies exclusively on a model but also on information stemming from measurements.

Process control and optimization in manufacturing is divided into five levels [93]. This is shown in Figure 3.18 with the time scales for each level. It is to be noted here that RTO forms part of level 4 and contrary to presumptions [94], is not continuous process like measurements. In fact, RTO is executed in an hourly or even daily interval depending on the change in input measurements, which are continuous.

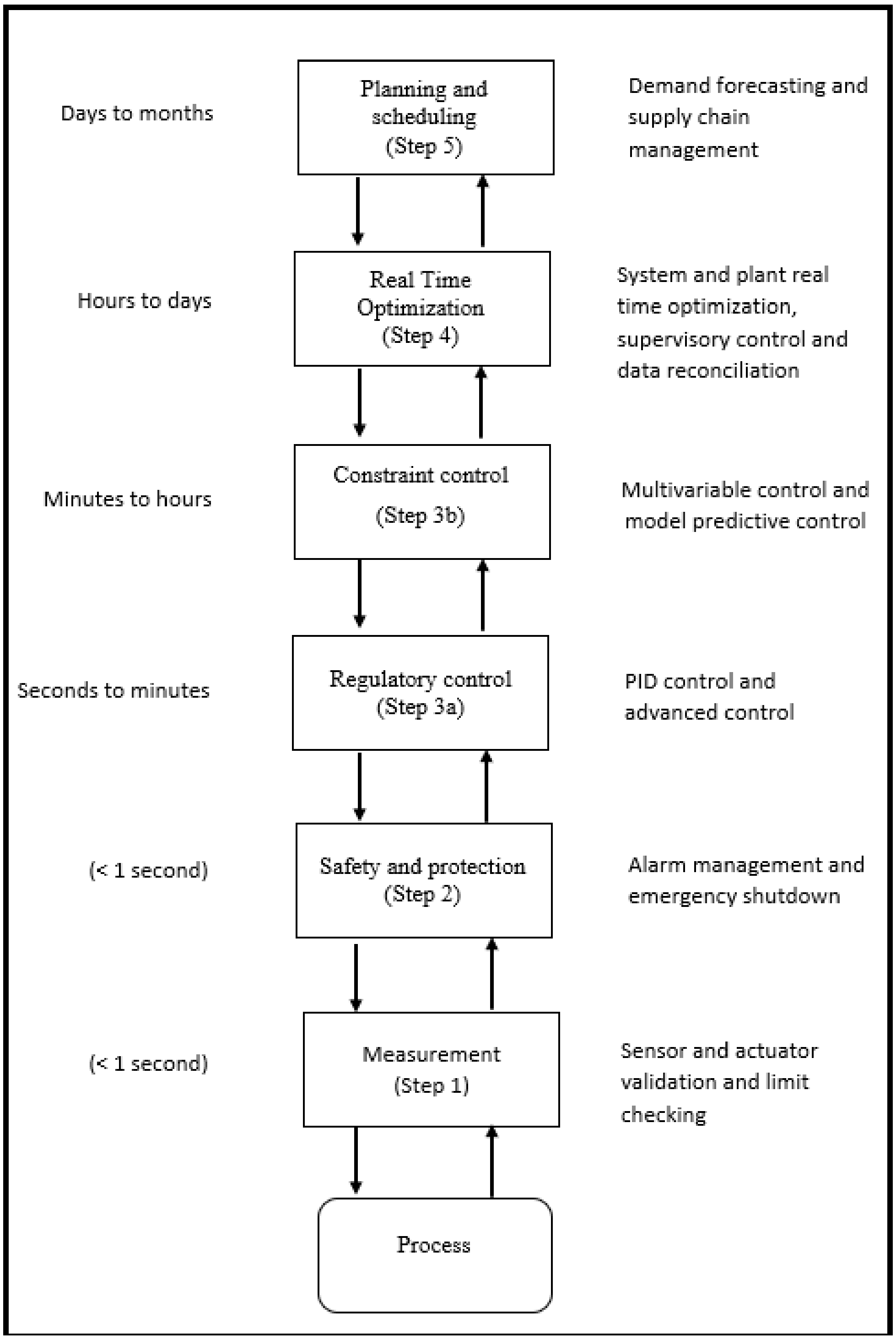

Days to months
Planning and scheduling (Step 5)
Demand forecasting and supply chain management
Hours to days
Real Time Optimization (Step 4)
System and plant real time optimization, supervisory control and data reconciliation
Minutes to hours
Constraint control (Step 3b)
Multivariable control and model predictive control
Seconds to minutes
Regulatory control (Step 3a)
PID control and advanced control
(< 1 second)
Safety and protection (Step 2)
Alarm management and emergency shutdown
(< 1 second)
Measurement (Step 1)
Sensor and actuator validation and limit checking
Process

Figure 3. 18 The five levels of process control and optimization in manufacturing with timescales for each stage [93]

3.6.3.1 Architecture of Real Time Optimization

The previous section established the position and time scale of RTO in process control and optimization. This section looks at the architecture of an RTO [95] in a closed loop control system. The architecture is shown in Figure 3.19.

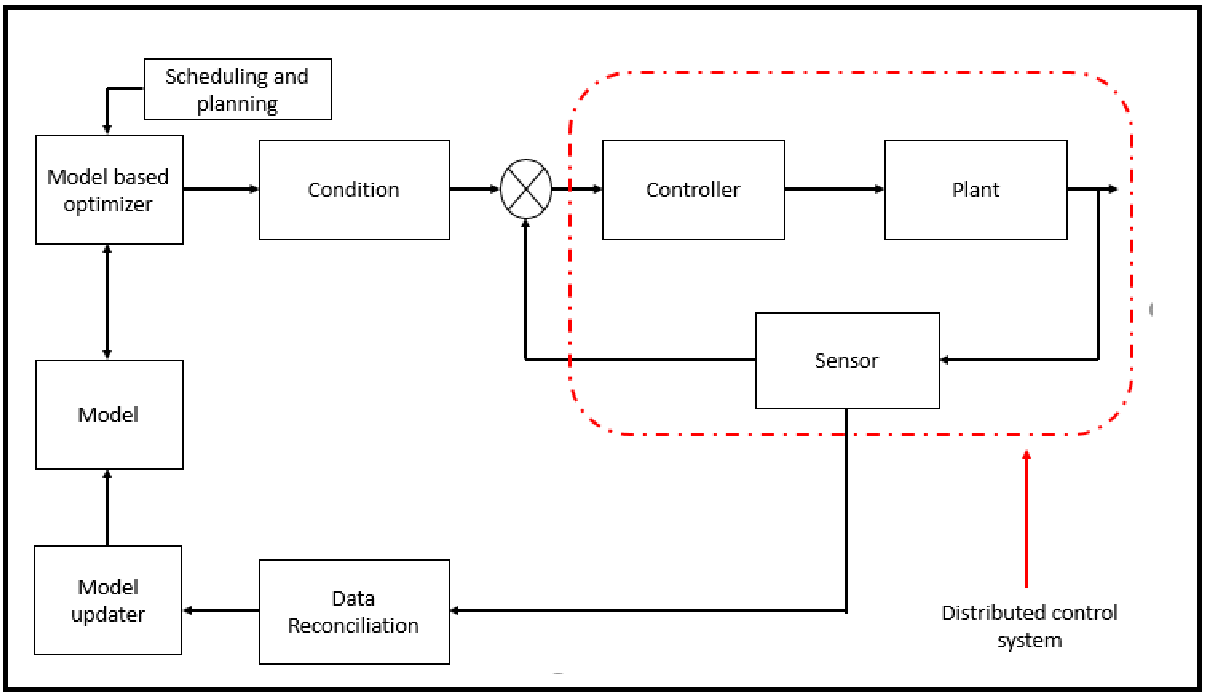


Figure 3. 19 Architecture of Real Time Optimization

As seen in Figure 3.19, the architecture of a RTO system consists of a Distributed Control System (DCS) which executes the model. The output of the sensor in the DCS is fed to data reconciliation block. Data reconciliation is the process of identifying and improving data that is corrupted by the errors during measurement and transmission process.

The reconciled data is send to the model updater, which contains the process parameters. Once the model is updated, it is send to the model based optimizer where model based optimization is performed with inputs from the scheduling and planning section.

The main applications of real time optimization are as follows;

- Error elimination
- Optimizing plant performance
- Performing fault detection

- Assessing the energy consumption of the plant
- Calculating the consumption rate of raw materials used in production
- Sending computer data to SCADA system

3.6.3.2 Formulating and Solving a Real Time Optimization problem

The following steps are used in performing a RTO [96], these steps will be used later in the section to formulate the optimization technique for the customized plant model that was discussed in Section 3.4.

1. Determine the process variables of interest
2. Define the objective function
   a. Determine the optimization criterion
3. Development of process models
   a. Define equality constraints related to process variables
   b. Define inequality constraints related to physical structure of the model.
   c. Determine a mathematical expression which describes the model based on the equalities and inequalities.
4. Simplify the process model
   a. Ignore process variables which have negligible effect on the objective function.
5. Apply a suitable optimization technique
   a. Align the problem to an equation that was discussed in Section 3.5.2.4 based on the relationships between the process variables and the objective function.
   b. Choose a platform or software to optimize the equation based on the criteria stipulated in 5a.
6. Check for sensitivity
   a. Analyse if the model is sensitive to change in certain parameters in the model.

3.6.4 Formulating the Plant Model as an Optimization Problem

The steps described in Section 3.5.3.2 can be used formulate optimization problem for the plant developed in Section 3.4.

- Step 1: Determine process variables – The plant model has the following variables;
    - Water stored in the tank in the Source subsystem
    - Flow rate of water from the pump in the storage tank subsystem
    - Initial number of 500ml bottles in the bottle manufacturing subsystem
    - Initial number of 750ml bottles in the bottle manufacturing subsystem
    - Expected date of delivery of customer orders
- Step 2: Defining the objective function – The objective function of the plant model is to reduce the production time for completing the customer orders. The hypothesis is that with optimization the production time can be significantly improved.
- Step 3: Development of process models - The model has considered two constraints being firstly the water level of the tank, defined in equation 3.3, and secondly the number of 500ml and 750ml bottles available in storage. The water level in the tank should never go below 0% and should create an alert when below 25%. The number of bottles in storage should never go below zero as this would result in the system crashing in a physical setup.
- Step 4: Simplify the process model – In order to simplify the process, the initial number of bottles is kept above zero. The pump flow rate from the storage tank subsystem acts as the handle which can be varied to meet the constraints.
- Step 5: Apply a suitable optimization technique – On analysing the objective function, process variables and the constraints, it is noted that they exhibit a nonlinear relationship. Since the modelling is done on Simulink, the optimization can be done using MATLAB. MATLAB is widely used for optimization due to its flexibility [93] and ability to exchange data with other software packages. Section 3.5.5 will discuss the detail of the functions used in MATLAB.

- Step 6: Check for sensitivity – The sensitivity check will be discussed in Section 4 when unpacking the results.

3.6.5 Using MATLAB for Nonlinear Optimization

The Optimization ToolBox™ in MATLAB offers variety of functions to solve the different types of optimization problems discussed in Section 3.5.2. The scope of this study restricts the focus to Nonlinear optimization which on its own is a very broad topic. The discussions in this section are limited to focussing on analysing constrained nonlinear optimization techniques. There are mainly three functions available in MATLAB for nonlinear constrained optimization [97]. They along with their specific purpose is listed as follows;

- *fminbnd* – Finding minimum variable function on fixed intervals
- *fmincon* – Find minimum of constrained nonlinear multivariable function
- *fseminf* – Find minimum of semi-infinitely constrained multivariable nonlinear function

It is evident that in the listed MATLAB functions, *fmincon*, is best suited for the plant model optimization as it caters to minimizing nonlinear objective equations with constraints and multivariable functions. The *fmincon* function [97] has the following structure;

$$\boldsymbol{min_x\, f(x) such\ that} \begin{cases} \boldsymbol{c(x) \leq\ 0} \\ \boldsymbol{ceq\ (x) = 0} \\ \boldsymbol{A.x\ \leq b} \\ \boldsymbol{Aeq.x = beq} \\ \boldsymbol{lb \leq x \leq ub} \end{cases}$$

Where,

$b\ and\ beq\ are\ linear\ vectors$

$A\ and\ Aeq\ are\ matrices$

$c(x) and\ ceq(x) are\ non\ linearfunctions\ that\ return\ vectors$

$lb = lower\ boundary\ of\ constraint$

$ub = upper\ boundary\ of\ constraint$

$f(x) = function\ that\ needs\ to\ minimized\ returning\ a\ scalar$

3.6.6 Plant Model Optimization Syntax Using *fmincon*

As discussed in step 4 in Section 3.5.4, the plant model will use the pump flow rate from the storage tank subsystem as the handle which can be varied to meet the constraints described in step 3. The pump flow rate is defined under a mask in the storage tank subsystem as a variable 'x' so that it can be varied. This is shown in Figure 3.7.

The syntax used in *fmincon* to achieve this optimization is described as follows [97];

$x = fmincon(fun, x0, A, b, Aeq, Beq, lb, ub, nonlcon, opt)$

Where,

$x0 = strating\ point\ of\ minimization$

$fun = function\ to\ minimize$

$b\ and\ beq\ are\ vectors$ linear constraints

$A\ and\ Aeq\ are\ matrices$

$lb = lower\ boundary\ of\ constraint$

$ub = upper\ boundary\ of\ constraint$

Here, the syntax needs $x0$, a starting point for the minimization. This is kept at 0.1. Next it needs a lower and upper boundary for the pump flow rate, which is the handle. This will ensure that the solution is always within the range of $lb \leq x \leq ub$. The $lb =\ 0$ and the $ub = 1$. As there are no inequality constraints $Aeq = [\,]$, and $beq = [\,]$.

Therefore, the optimization should yield a pump flow rate between 0.1 and 1.0. Ideally the pump flow rate should be closer to 1.0, but that would drain the water in tank faster and use up bottles quicker, so the pump flow rate needs to varied to ensure both constraints are within bounds.

The results of the optimization will be discussed in Section 4.

Chapter 4 Results and Analysis

4.1 Introduction

The aim of this chapter is to showcase the tests that were done on the model and the results of the simulations that were obtained after the tests. The results are shared and analyzed in the same order as the modelling, discussed in Chapter 3. Initially, tests done on the primary model and the subsequent results thereof are shared. Next, tests and results of the customized model with inputs from the cloud server are analyzed. Finally, the tests and results of the model optimization is shown.

4.2 Primary Model Testing and Results

The primary model of the plant was discussed in Section 3.3. The model was divided into five subsystems discussed in Sections 3.3.1 to 3.3.5. As explained and depicted in Figures 3.5, 3.7, 3.9 and 3.11, the subsystems have masked values which determine the working of the model. The main aim of this model is to ascertain that water can be filled in 500ml and 750ml bottles in batches of six without depleting the bottle inventory. The subsystem mask values used for this testing is shown in Table 1.

Table 4. 1 Subsystem mask vales

Subsystem	Quantity	Mask value
Source	Source flow rate	8kg/sec
Source	Source upper limit	10kg/sec
Storage Tank	Pump flow rate	4kg/sec
Storage Tank	Pump upper limit	10kg/sec
Storage Tank	Maximum Tank capacity	1000m3
Storage Tank	Initial tank capacity	0m3
Bottle manufacturing	Time taken to produce bottle	2sec
Bottle storage	Initial number of 500ml bottles	0
Bottle storage	Initial number of 750ml bottles	0
Bottle storage	Output rate of bottles	2sec

As seen from the mask values, the pump flow rate (4kg/sec) is kept at half the source flow rate (8kg/sec). This will ensure that the water in the source tank does not get depleted as fast as the water in the storage tank. Another important mask value, is the initial number of 500ml and 750ml bottles. Both are placed at zero with each batch of bottles being filled in 2 seconds.

The first set of tests are done to determine if bottles can be filled in batches of six and the rate of production. This set of results are derived from the water filling subsystem described in Section 3.3.5. The result of this test is shown in Figure 4.1.

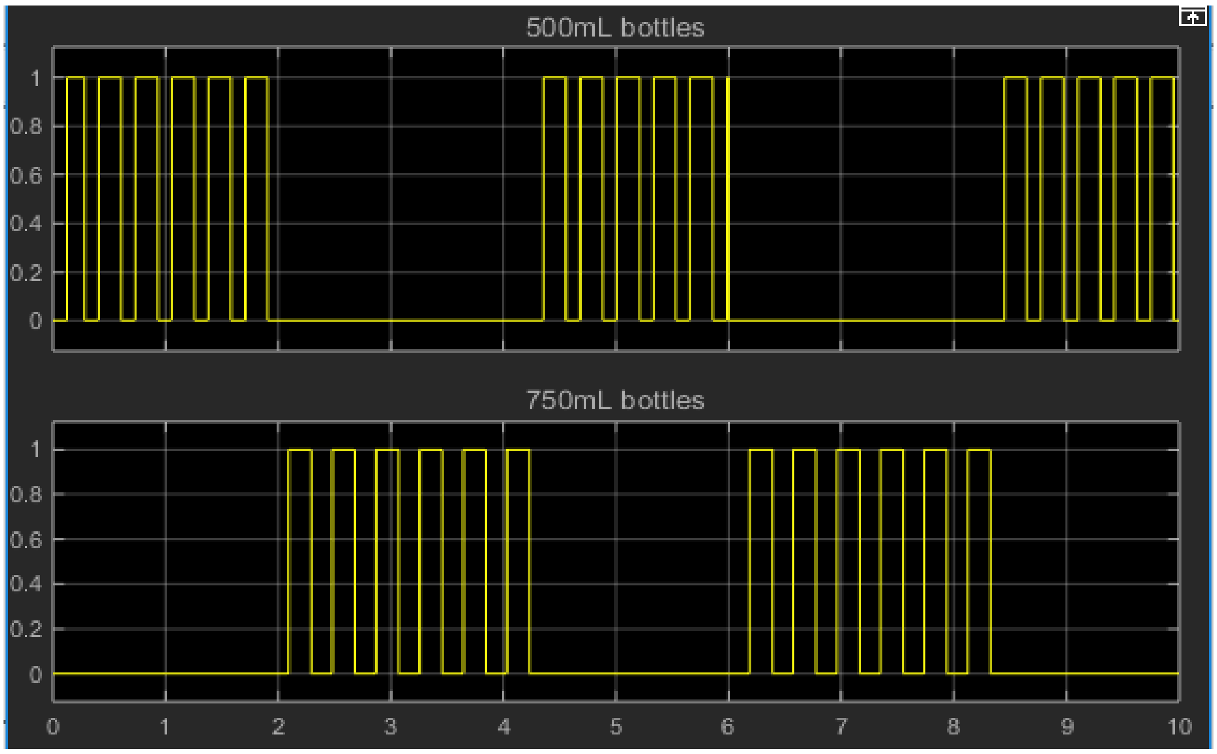


Figure 4. 1 Simulated output of water filling subsystem of primary model

It can be seen from Figure 4.1 that there is a clear distinction without overlap between the 500ml and 750ml bottles. The six spikes symbolize the six bottles in a batch, with each batch being produced in 2 seconds as stipulated in the subsystem mask.

The next set of tests focus on the number of bottles remaining in the storage after the bottles have been filled. This set of results are obtained from the bottle storage subsystem described in Section 3.3.4. The result is shown in Figure 4.2.

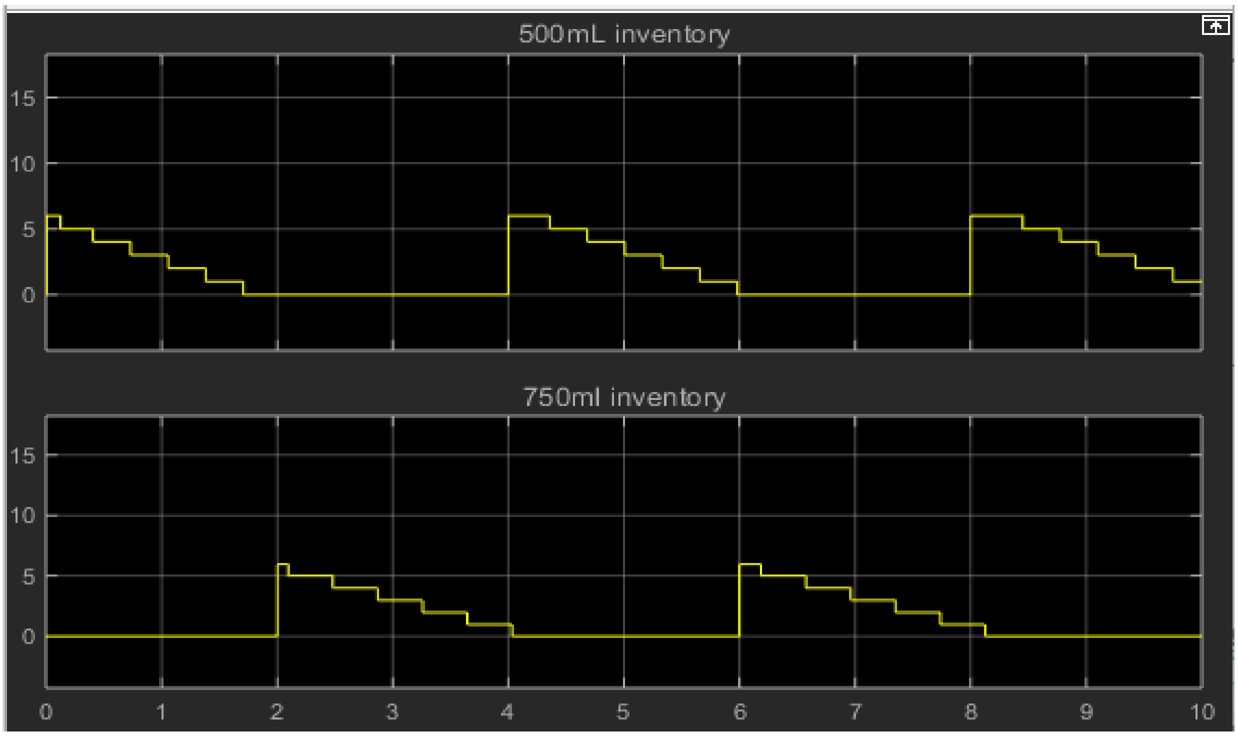


Figure 4. 2 Simulated output of bottle storage subsystem of primary model

It can be seen from Figure 4.2 that the bottle inventory never goes below zero during the production. This is a very important point to note because if the inventory goes below zero, it would mean that the plant would crash in a physical setup. It needs to be noted that this inventory level is achieved with no bottles in the inventory before the start of the process.

The last set of tests on the primary model is to determine if the model can make a distinction between 500ml and 750ml bottles. The results of this test is critical as it determines if the model can exhibit properties of a multi-mixed model assembly line. The result is obtained from the water filling subsystem and shown in Figure 4.3

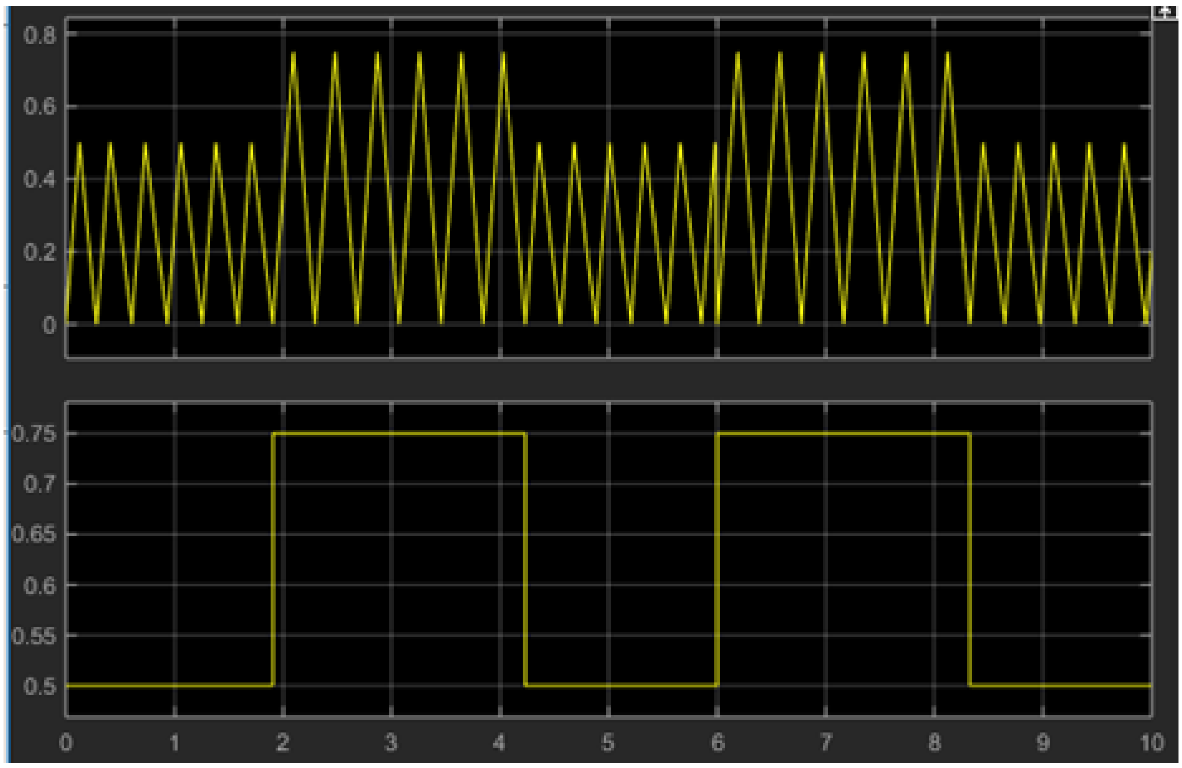


Figure 4. 3 Simulated output showing distinction between 500ml and 750ml bottles

The simulated output in Figure 4.3 shows six alternating spikes occurring every two seconds. The six spikes indicate a batch of bottles and the two seconds is defined in the mask as the manufacturing time for each batch. As can be seen in Figure 4.3, there is clear distinction between the 500ml and 750ml bottles as indicated by the difference in the amplitude of the spikes.

The aim of designing the primary model was to ascertain that a multi mixed assembly could be developed. The simulated output of the water filling subsystem displayed in Figure 4.3 is evidence that this is indeed possible. The other results from the water filling subsystem show how the bottle is filled with water in alternating cycles. This is displayed in Figure 4.1. The status of the bottle inventory during this continuous process is also shown in Figure 4.2. It is to be noted that there occurs no overlap of bottles in Figure 4.1 and the bottle storage is not exhausted at any point during the production.

Although the results shown in all Figures in this section are only for 10 seconds, actual simulations were done for 24 hours and yielded very similar results, especially with respect to the bottle inventory which was always above zero. However, this set of results are for a specific set of subsystem mask values displayed in Table 2. This means the model is not customized. The next section focuses on the model customization.

4.3 Customized Model Testing and Results

This section discusses how the model was customized to function as make-to-order assembly line. This is done in two parts. Firstly, the primary model was redesigned to accept manual inputs. Secondly, the redesigned model was linked to a web app, so as to make ordering completely automated.

The results of the primary model discussed in Section 4.2 show that the bottles can be filled in alternate batches of six. This is due to the design of water filling subsystem, discussed in Section 3.3.5. In order to customize the model, water filling subsystem is redesigned to make use of a customer requirements subsystem. The detailed working of the subsystem is described in Section 3.4 and depicted in Figure 3.14.

The customer requirements subsystem was tested using the inputs shown in Table 2.

Table 4. 2 Customer requirements table

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date of delivery
A	90	90	09-March-2019
B	125	60	08-March-2019
C	60	150	11-March 2019
TOTAL	275	300	

The model, firstly, sorts the orders according to date of delivery with the earliest date of delivery being processed first. Therefore, the table that is processed by the model looks as shown in Table 3.

Table 4. 3 Customer requirements table sorted according to date of delivery

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date of delivery
B	125	60	08-March-2019
A	90	90	09-March-2019
C	60	150	11-March 2019
TOTAL	275	300	

The resulting scope output from the water filling subsystem is shown in Figure 4.4.

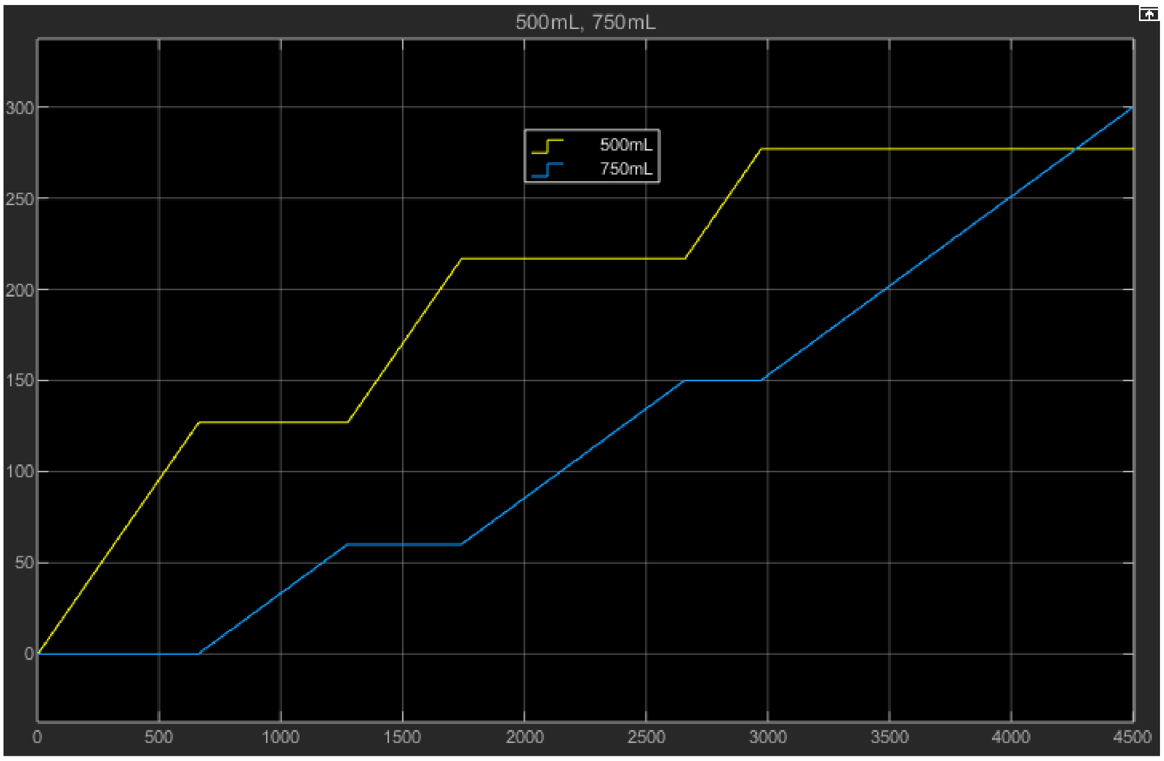


Figure 4. 4 Simulated output of water filling subsystem for the customized model

As seen from Figure 4.4, the order by customer B is completed first, followed by customer A and C. This is in accordance with Table 2. The scope output of the bottle storage for the same input is depicted in Figure 4.5.

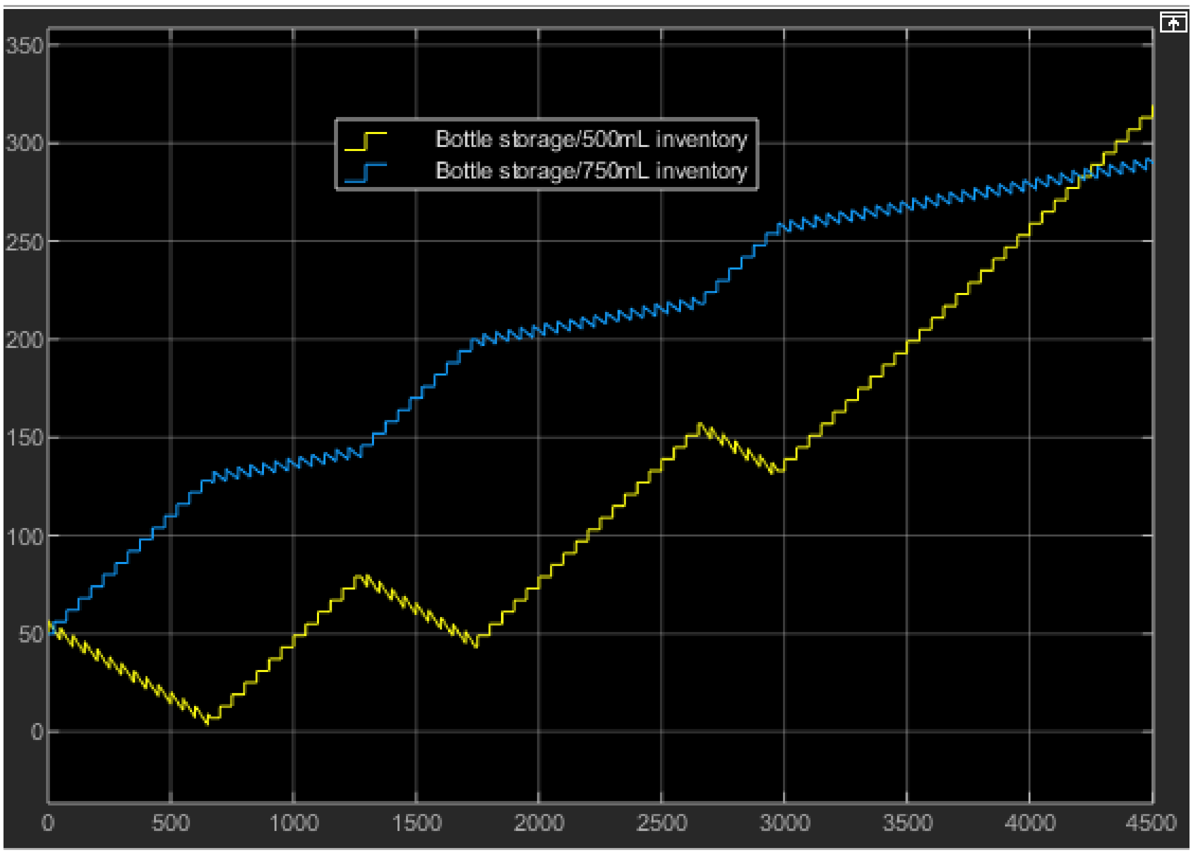


Figure 4. 5 Simulated output of bottle storage subsystem of the customized model

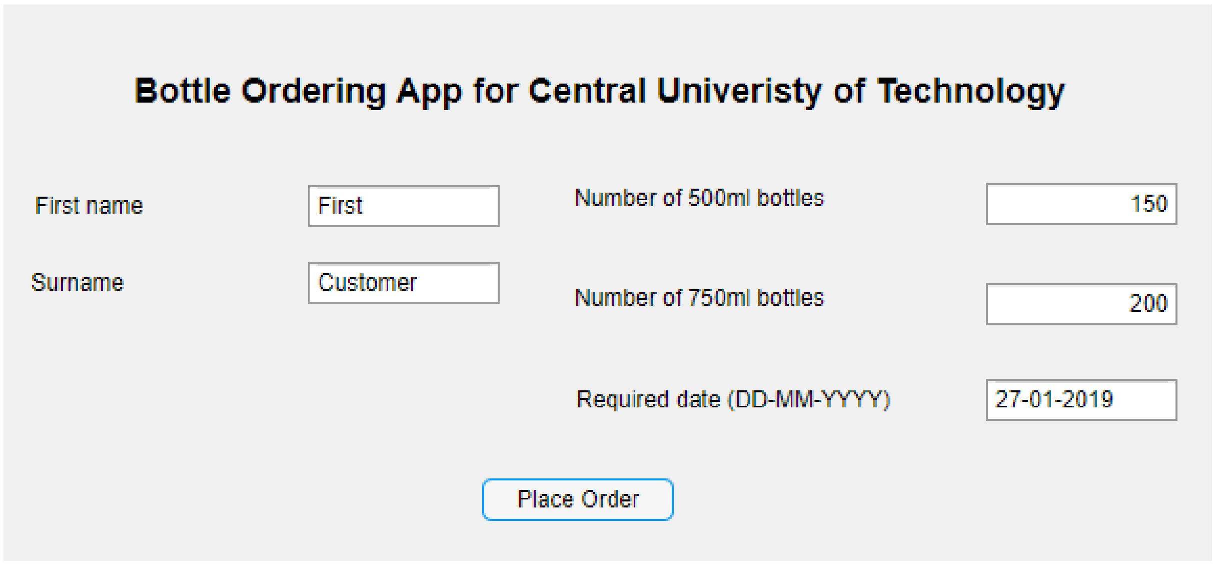
Figure 4.5 shows that the inventory for both 500ml and 750ml bottles are above zero at all times during the water filling process. This is critical, as mentioned previously, as it ensures that the model does not crash in real time operation.

On analyzing and comparing Figure 4.1 and 4.4, it can be seen that there is an improvement in the design of the model. The model has gone from one which outputs 500ml and 750ml bottles in batches of six to one which can manually take inputs from customers and output them accordingly.

The robustness of the model is apparent when comparing Figure 4.2 and 4.5. The bottle inventory of the primary model gets depleted after the production of each batch, but in the customized model, the bottle inventory is always above the threshold even after filling over 500 combined bottles of water.

However, the model is not fully automated as every new order has to be entered physically into the model and the old orders have to be deleted. Therefore, the next step in the design of the model is to make model into a complete make-to-order system.

The design of a make-to-order system was discussed in Section 3.5 and was accomplished using a web app developed using MATLAB and hosted on a cloud server. A GUI, which is used to capture the customer requirements is shown in Figure 4.6.

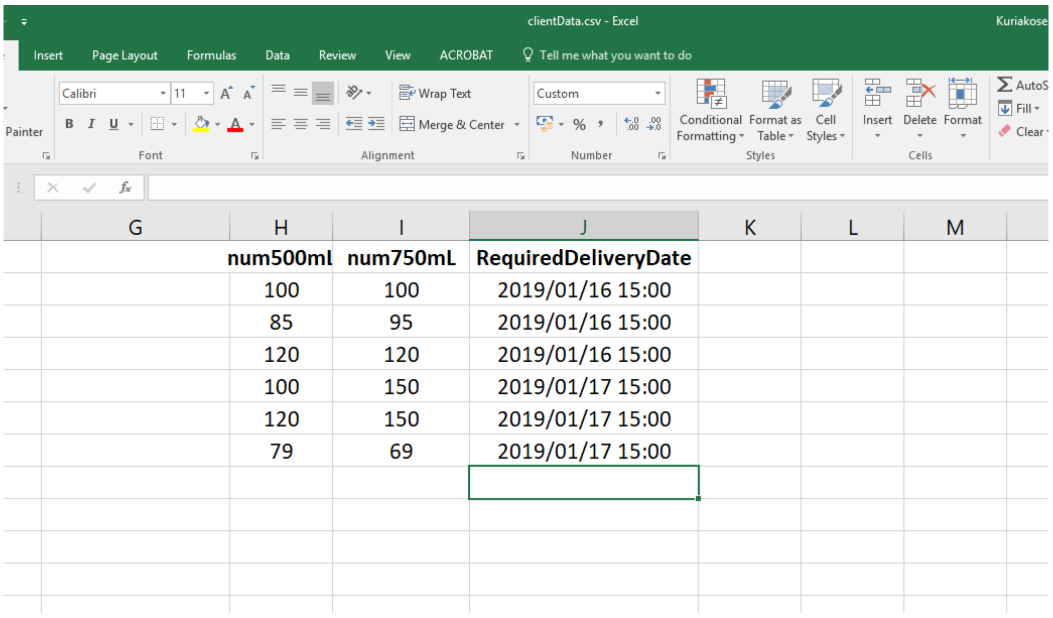

Bottle Ordering App for Central Univeristy of Technology

First name	First	Number of 500ml bottles	150
Surname	Customer	Number of 750ml bottles	200
		Required date (DD-MM-YYYY)	27-01-2019

Figure 4. 6 A GUI used by customers to place bottle orders

The orders placed using the GUI are captured in an excel sheet and saved on the server machine. A typical order sheet containing the customer requirements in shown in Figure 4.7. It is noticeable here that the orders are captured and appended on to the excel sheet and sorted according to the priority in which the orders need to be executed.

Once a cycle of orders is completed, the excel sheet is passed on to MATLAB for optimization. Upon completion of optimization, the order sheet is deleted of its contents and awaits a fresh set of orders.



num500ml	num750mL	RequiredDeliveryDate
100	100	2019/01/16 15:00
85	95	2019/01/16 15:00
120	120	2019/01/16 15:00
100	150	2019/01/17 15:00
120	150	2019/01/17 15:00
79	69	2019/01/17 15:00

Figure 4. 7 The excel sheet containing the customer requirements

4.4 Non-Optimized Operation of the Customized Make-to-Order Model

This section focuses on the results of the non-optimized operation of the make to order model that was discussed in the previous section. The aim of the section is to assess the functioning of the model without optimization. This would assist in proving the limitations of the existing research discussed in Section 2.5.

The customer requirements contained in the excel sheet shown in Figure 4.7 is pulled in using a MATLAB file. This resulting table is depicted in Table 4. It is to be noted here that Table 4 only contains the customer requirements and the required date and time of delivery.

The next MATLAB program appends two columns to table before optimization, the first columns showing the expected delivery date and time without optimization and the second column showing the expected delivery date and time after optimization of the production time. This is shown in Table 5. It should be noted that the two columns do not contain any data as the optimization process has not started.

Table 4. 4 Customer requirements table as per the input from cloud server

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery
1	100	100	16-Jan-2019 15:00
2	85	95	16-Jan-2019 15:00
3	120	120	16-Jan-2019 15:00
4	100	150	17-Jan-2019 15:00
5	120	150	17-Jan-2019 15:00
6	79	69	17-Jan-2019 15:00
TOTAL	604	684	

Table 4. 5 Customer requirements table with appended columns

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery	Non optimized date and time of delivery	Optimized date and time of delivery
1	100	100	16-Jan-2019 15:00	NaT	NaT
2	85	95	16-Jan-2019 15:00	NaT	NaT
3	120	120	16-Jan-2019 15:00	NaT	NaT
4	100	150	17-Jan-2019 15:00	NaT	NaT
5	120	150	17-Jan-2019 15:00	NaT	NaT
6	79	69	17-Jan-2019 15:00	NaT	NaT
TOTAL	604	684			

Initially, the model is provided with a random, non-optimized pump flow rate. This will give an approximate date of completion of the orders. This results in the completion of the first appended column with non-optimized date and time of delivery and is shown in Table 6.

Table 4. 6 Customer requirements table with non-optimized date of delivery with water flow rate at 0.6m/sec

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery	Non optimized date and time of delivery
1	100	100	16-Jan-2019 15:00	16-Jan-2019 02:54
2	85	95	16-Jan-2019 15:00	16-Jan-2019 03:14
3	120	120	16-Jan-2019 15:00	16-Jan-2019 03:42
4	100	150	17-Jan-2019 15:00	17-Jan-2019 12:07
5	120	150	17-Jan-2019 15:00	17-Jan-2019 16:37
6	79	69	17-Jan-2019 15:00	17-Jan-2019 17:01
TOTAL	604	684		

It can be deduced from Table 6 that first four orders (Customer's 1, 2, 3 and 4) are completed before the required date of delivery. However, the last two orders (Customer's 5 and 6) are not met. This is owing to the fact that the high water flow rate has resulted in the water in the tank being depleted. Therefore, the assembly line process has to be halted to allow for the tank to be replenished. This is indicative of negative drift, described in Section 2.3.2 and depicted in Figure 2.4.

After the tank has been replenished, the assembly line process continues, but the time lost during the replenishing cannot be made up and hence the customer orders 5 and 6 fall behind of schedule. This describes an instance of positive drift. A GUI depicting the condition of the constraints at the end of the process is shown in Figure 4.8.

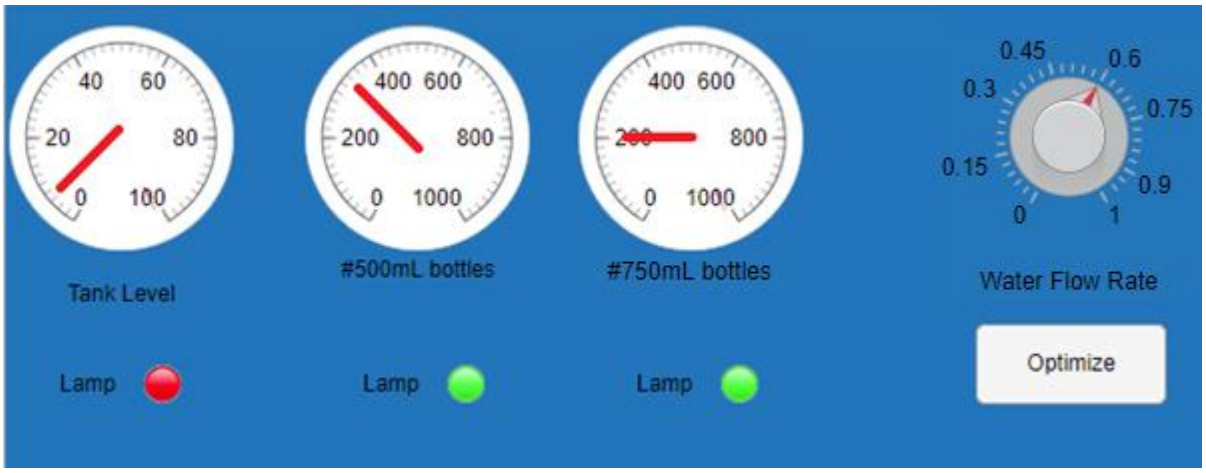


Figure 4. 8 GUI showing status of constraints and handle at 0.6m/sec flow rate

The GUI shown in Figure 4.8 consists of three gauges which show the status of the constraints and a knob which can be used to control the flow rate. The constraints were defined in Step 3 of Section 3.6.4 under the problem formulation. The LED indicators show if the constraint has been met with a green light and a red light indicates that a constraint has not been met. It is evident from Figure 4.8 that at the end of the process the Tank level is below the required 25%, while the number of 500ml and 750ml bottles are above the required level.

To overcome this problem, the water flow rate was reduced to 0.1 and the tests were conducted once again. The results are shown in Table 7. It can be seen here that the rate of completing the orders is much slower in this instance and customer orders 3 and 6 are not completed within the required time, pointing to an instance of positive drift. The status of the constraints is depicted in Figure 4.9. As a result of the slower water flow rate, all constraints are met.

Table 4. 7 Customer requirements table with non-optimized date of delivery with water flow at 0.1m/s

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery	Non optimized date and time of delivery
1	100	100	16-Jan-2019 15:00	16-Jan-2019 12:54
2	85	95	16-Jan-2019 15:00	16-Jan-2019 13:14
3	120	120	16-Jan-2019 15:00	16-Jan-2019 21:42
4	100	150	17-Jan-2019 15:00	17-Jan-2019 07:07
5	120	150	17-Jan-2019 15:00	17-Jan-2019 13:37
6	79	69	17-Jan-2019 15:00	17-Jan-2019 15:53
TOTAL	604	684		

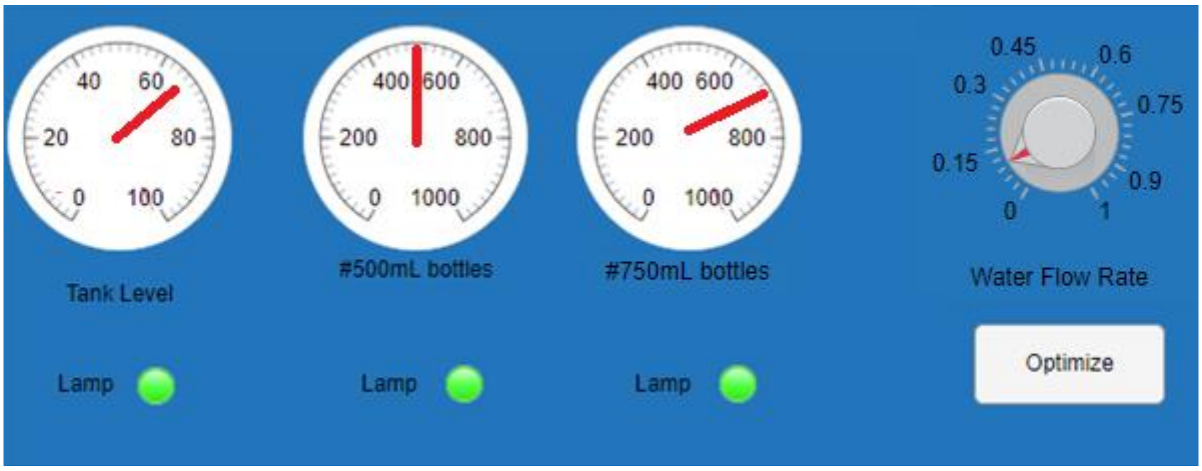


Figure 4. 9 GUI showing condition of constraints and handle at 0.1m/sec flow rate

The test results from Figures 4.8 and 4.9 and Table 6 and Table 7 prove how drift, especially positive drift, becomes a challenge in an assembly line. Several other tests with different inputs were done to understand the extend of this problem. It was seen that the instances of positive drift increased with the increase in orders and constraints. The challenge was also accentuated if the required delivery dates were close to each other.

These tests and results put a strong emphasis for the need for a robust optimization function for this model. The optimization should be able to meet all constraints while ensuring that the required delivery dates are met. The MATLAB optimization function *fmincon*, discussed in depth in Section 3.6.6, is used in this research. The results of the optimization are discussed in Section 4.5

4.5 Optimized Operation of the Customized Make-To-Order Model

The tests documented in Section 4.4, show the need for optimization. As mentioned in Section 4.4, MATLAB optimization function *fmincon* is used in this study. For continuity purposes, the same set of inputs used in Section 4.4, described in Table 4.4, is used to describe the optimization process.

The syntax for the *fmincon* function is described in Section 3.6.6 is as follows

$x = fmincon(fun, x0, A, b, Aeq, Beq, lb, ub, nonlcon, opt)$

Here,

$x0 = strating\ point\ of\ minimization$

$fun = function\ to\ minimize$

$b\ and\ beq\ are\ vectors$ linear constraints

$A\ and\ Aeq\ are\ matrices$

$lb = lower\ boundary\ of\ constraint$

$ub = upper\ boundary\ of\ constraint$

In this specific optimization example, the syntax is as follows

$[xOpt, TTMOpt] = fmincon(fun, [0.1; 0.1], [\,], [\,], [\,], [\,], [0; 0], [1; 1], funConstr, opt)$

Here, the starting point of the pump flow rate, which is the constraint, and the lower and upper boundaries of the constraint are defined. The starting point is defined as 0.1 and the lower boundary is 0, while the upper boundary is 1. *XOpt* is the optimized pump flow rate which will meet the defined constraints and *TTMOpt* is the 'Time To Manufacture' with optimized pump flow rate.

After completing the optimization, the *fmincon* function does a further check to test the robustness of the model by adding 0.001 to the optimized pump flow rate. The check should meet the constraints as before to ensure the model is robust and not prone to even the minutest of changes. The results of the test are shown in Table 8. It is to be noted here that the non-optimized date of delivery is kept at 0.1m/sec.

The optimized pump flow rate for this specific set of inputs was 0.36. It can be seen here that at the optimized pump flow rate all the customer orders are met well before the required date and time of delivery. On further analysis, the constraints such as the level of water in the tank and the number of bottles available in the inventory are met on completion of the orders. The constraints are depicted in Figure 4.10

Table 4. 8 Customer requirements table with optimized date of delivery at 0.36m/s

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery	Non optimized date and time of delivery	Optimized date and time of delivery
1	100	100	16-Jan-2019 15:00	16-Jan-2019 12:54	16-Jan-2019 02:54
2	85	95	16-Jan-2019 15:00	16-Jan-2019 13:14	16-Jan-2019 03:30
3	120	120	16-Jan-2019 15:00	16-Jan-2019 21:42	16-Jan-2019 04:26
4	100	150	17-Jan-2019 15:00	17-Jan-2019 07:07	16-Jan-2019 05:17
5	120	150	17-Jan-2019 15:00	17-Jan-2019 13:37	16-Jan-2019 06:42
6	79	69	17-Jan-2019 15:00	17-Jan-2019 15:53	16-Jan-2019 08:53
TOTAL	604	684			

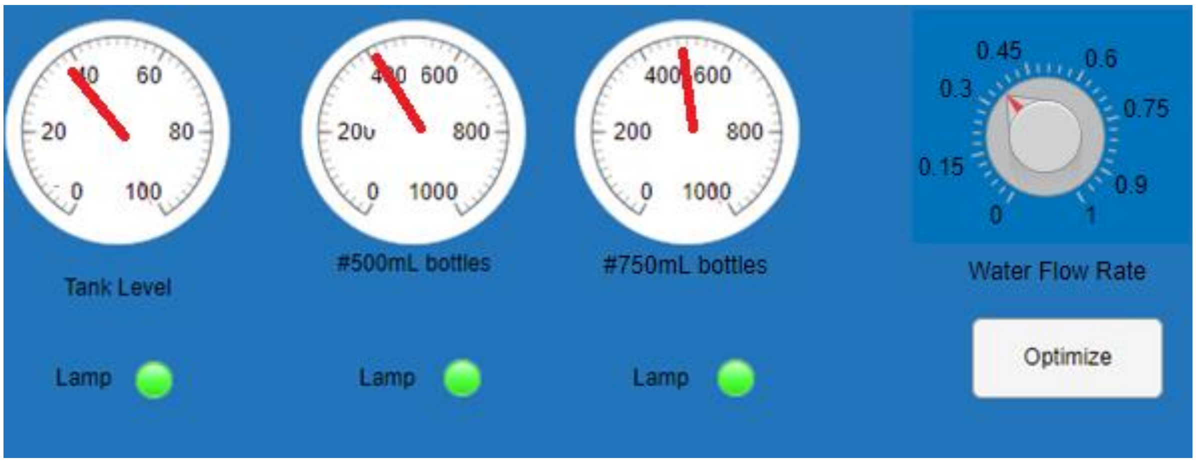


Figure 4. 10 GUI showing condition of constraints after optimization.

A graphical comparison between the optimized and non-optimized date and time of delivery is shown in Figure 4.11. It can be see here that in all instances the optimized date and time of delivery is less than the non-optimized date and time of delivery.

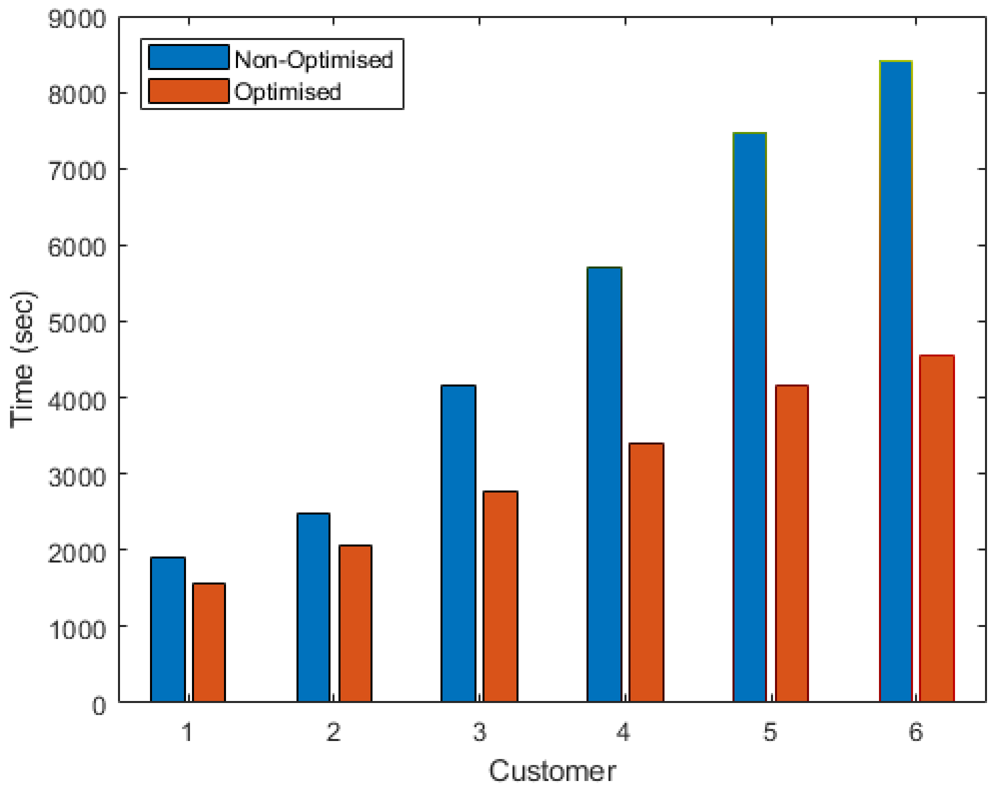


Figure 4. 11 Comparison between the optimized and non-optimized date and time of delivery.

To establish the veracity and robustness of the model, numerous sets of customer requirements with varying required delivery date and time were provided to the model. The results of the optimization are shown as comparison graphs, similar to that in Figure 4.11.

In the first instance, there are eleven customer inputs which need to be completed on same date at 15H00. The inputs are shown in Table 4.9

Table 4. 9 Customer requirements table with eleven inputs and same day required delivery

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery
1	100	100	2019/06/16 15:00
2	85	95	2019/06/16 15:00
3	120	120	2019/06/16 15:00
4	100	150	2019/06/16 15:00
5	120	150	2019/06/16 15:00
6	79	69	2019/06/16 15:00
7	25	30	2019/06/16 15:00
8	25	30	2019/06/16 15:00
9	45	50	2019/06/16 15:00
10	100	200	2019/06/16 15:00
11	150	250	2019/06/16 15:00

The results of the optimization are shown in Figure 4.12. It is evident here that the required delivery date has been met on all instances and furthermore, the optimized time of delivery is quicker for all inputs.

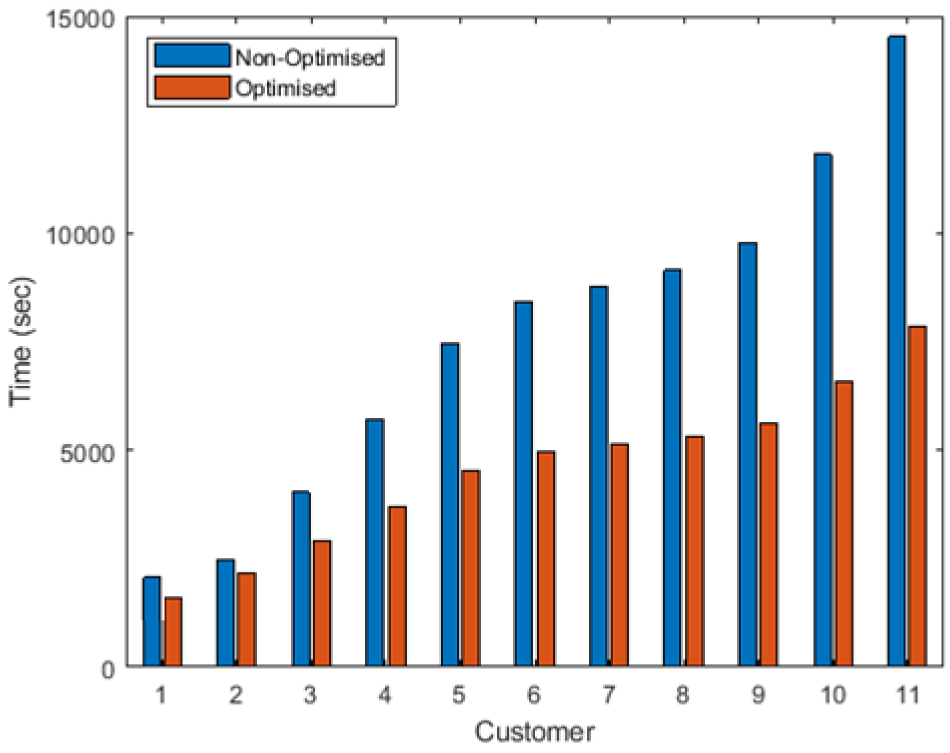


Figure 4. 12 Comparison between the optimized and non-optimized date of delivery for 11 inputs required on the same day and same time

In the second instance, eleven sets of inputs were provided with required delivery dates 2 hours apart from each other. In this instance the customer inputs were decreasing with each subsequent customer. Table 4.10 shows the inputs.

Table 4. 10 Customer requirements table with eleven inputs and required delivery dates two hours apart

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery
1	110	100	2019/06/16 8:00
2	100	90	2019/06/16 10:00
3	90	80	2019/06/16 12:00
4	80	70	2019/06/16 14:00
5	70	60	2019/06/16 16:00
6	60	50	2019/06/16 18:00
7	50	40	2019/06/16 20:00
8	40	30	2019/06/16 22:00
9	30	20	2019/06/17 0:00
10	20	10	2019/06/17 2:00
11	10	5	2019/06/17 4:00

The results of the optimization are shown in Figure 4.13. It is evident that, like in the first instance, the required date and time of delivery was met on all inputs and the optimized date of delivery was faster than the non-optimized date of delivery.

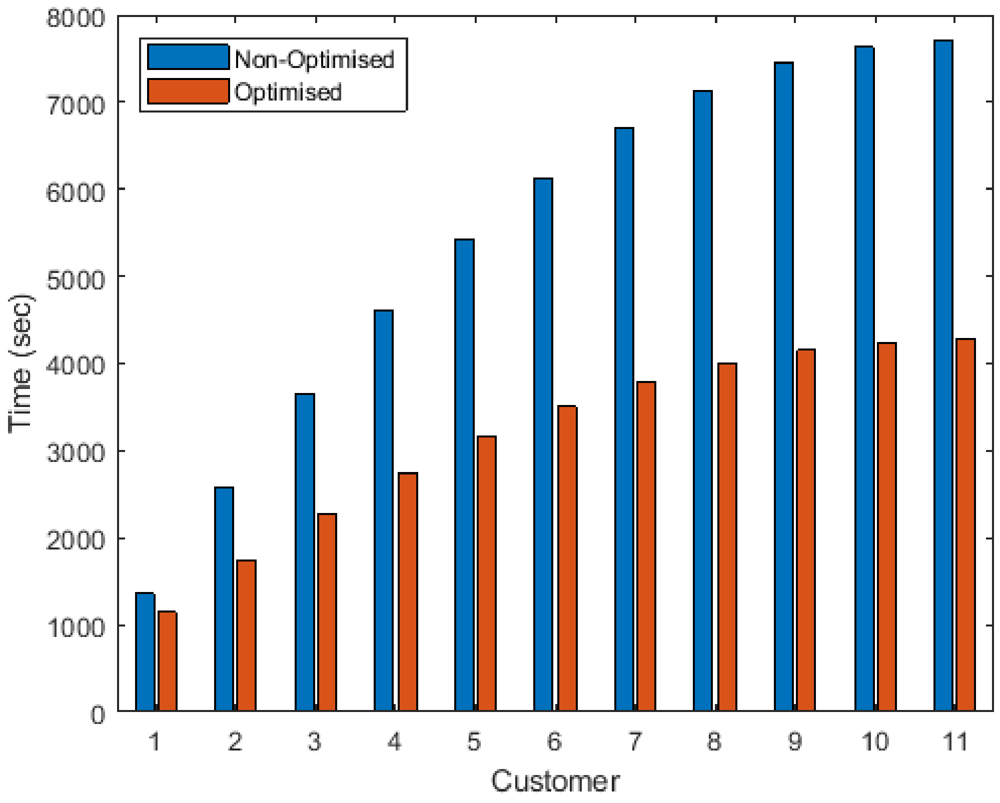


Figure 4. 13 Comparison between the optimized and non-optimized date of delivery for 11 inputs required on the same day two hours apart

In the third instance, eleven sets of customer inputs were provided with the required delivery date set for the same date as the requests were made. In this instance the customer inputs were initially decreasing and at a later stage increasing with each subsequent customer. This was done to examine how the model would handle a varying set of inputs. Table 4.11 shows the inputs.

Table 4. 11 Customer requirements table with ordering and required delivery date on same date

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery
1	100	100	2019/06/05 15:00
2	85	95	2019/06/05 15:00
3	120	120	2019/06/05 15:00
4	100	150	2019/06/05 15:00
5	120	150	2019/06/05 15:00
6	79	69	2019/06/05 15:00
7	25	30	2019/06/05 15:00
8	25	30	2019/06/05 15:00
9	45	50	2019/06/05 15:00
10	100	200	2019/06/05 15:00
11	150	250	2019/06/05 15:00

The results of the optimization are shown in Figure 4.14. As seen in the first two instances, the required delivery date is met in the third instance as well. However, on close examination it can be seen that the non-optimized delivery time for the first customer input is faster than the optimized delivery time in the third instance.

After the first customer input, there is a distinct difference between the optimized and non-optimized delivery time favoring the optimized delivery time. A closer look at the constraints while the first order is being executed shows that in the non-optimized state, the pump flow rate was very high, and this resulted in the water being filled quickly. At the same time the water in the tank was depleted, which meant that the subsequent orders were delayed.

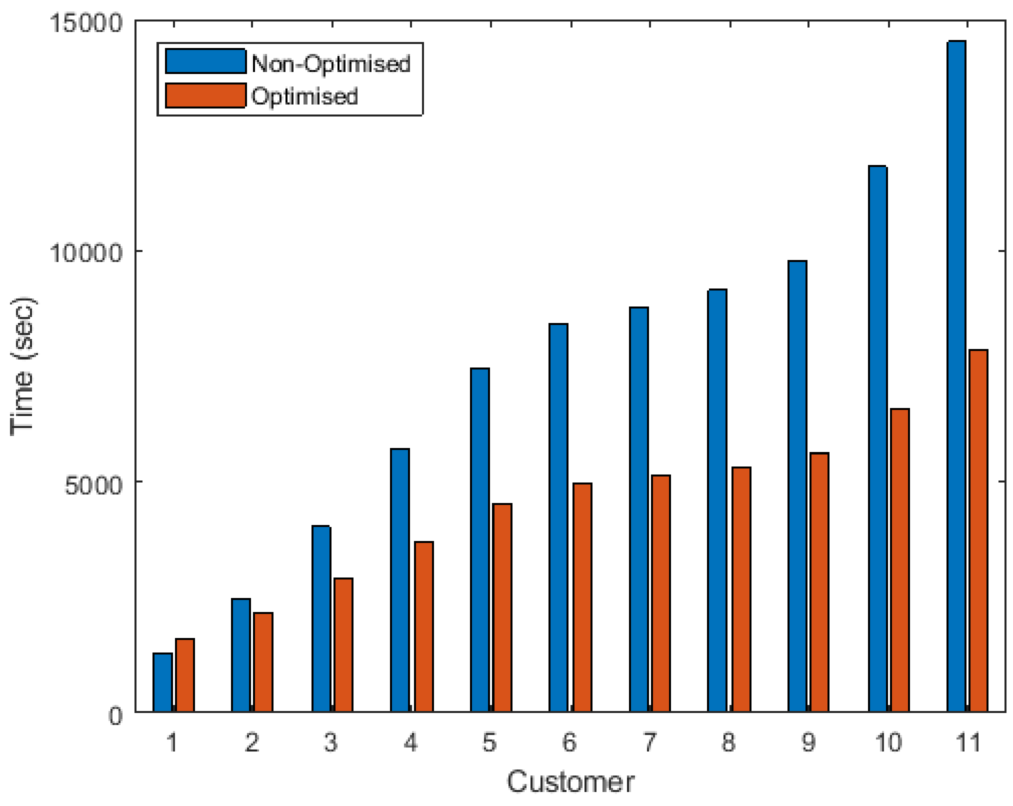


Figure 4. 14 Comparison between the optimized and non-optimized date of delivery for inputs orders and required delivery date on the same day

In the fourth instance, the impact of varying inputs was tested. Here, the number of 500ml orders were decreasing while the number of 750ml orders were increasing. Each subsequent order had a two-hour delay. Table 4.12 shows the customer inputs.

Table 4. 12 Customer requirements table with opposite customer gradients two hours apart

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery
1	100	50	2019/06/16 8:00
2	90	60	2019/06/16 10:00
3	80	70	2019/06/16 12:00
4	70	80	2019/06/16 14:00
5	60	90	2019/06/16 16:00
6	50	100	2019/06/16 18:00
7	40	110	2019/06/16 20:00
8	30	120	2019/06/16 22:00
9	20	130	2019/06/17 0:00
10	10	140	2019/06/17 2:00
11	5	150	2019/06/17 4:00

The results of the optimization are shown in Figure 4.15. As seen in all previous cases, the required delivery date has been met. However, the optimized delivery time of the first order is on par with the non-optimized delivery time. After the first order, all further requests are met with the optimized time of delivery faster than the non-optimized time of delivery. As in the previous instance, an examination of the status of the constraints showed that the pump flow rate was quite high during the execution of the first order. This explained why the first order was completed faster in the non-optimized state. However, like in the previous instance, this resulted in the water in the tank being depleted and therefore delay the subsequent customer orders.

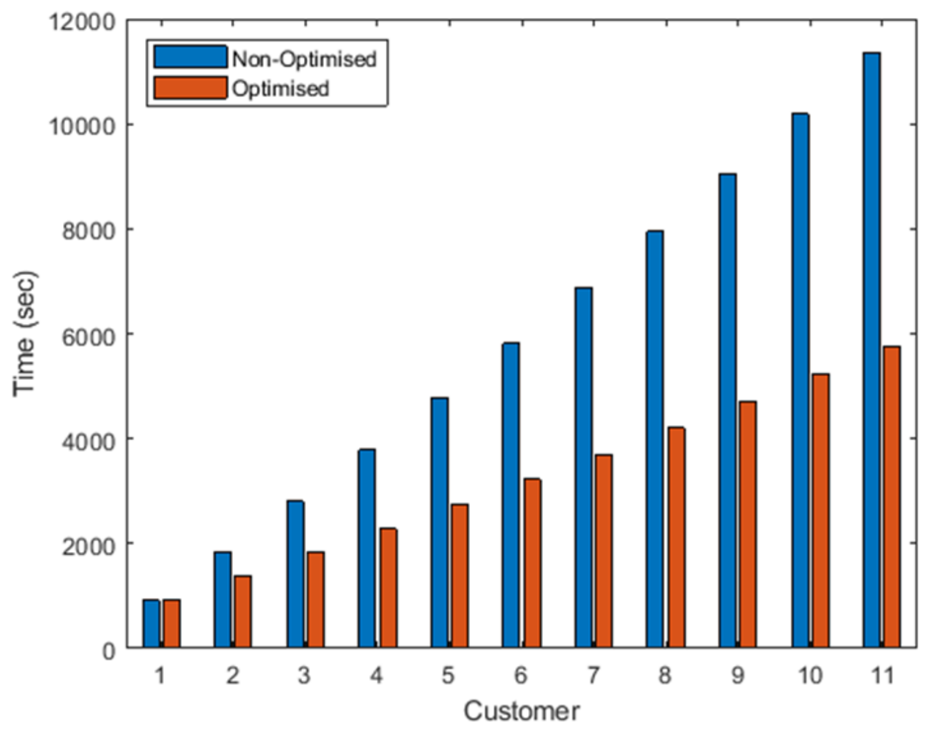


Figure 4. 15 Comparison between the optimized and non-optimized date of delivery for input orders with opposite gradients

In-order to test if the response depicted in Figure 4.15 was a result of the varying inputs, the model was input with a decreasing set of inputs two hours apart. This would be the fifth instance and is depicted in Table 4.13.

Table 4. 13 Customer requirements table with decreasing customer inputs two hours apart

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery
1	110	100	2019/06/16 8:00
2	100	90	2019/06/16 10:00
3	90	80	2019/06/16 12:00
4	80	70	2019/06/16 14:00
5	70	60	2019/06/16 16:00
6	60	50	2019/06/16 18:00
7	50	40	2019/06/16 20:00
8	40	30	2019/06/16 22:00
9	30	20	2019/06/17 0:00
10	20	10	2019/06/17 2:00
11	10	5	2019/06/17 4:00

The results of the optimization are shown in Figure 4.16. As in previous cases, the required delivery date has been met for all customers. Moreover, the anomaly that was seen in the previous instance has been overcome. In this instance, the optimized time of delivery is faster than the non-optimized date of delivery for all customer orders. It is also worth noting here that a similar instance of inputs with a 30-minute delay between subsequent orders was provided to the model and it yielded very similar results.

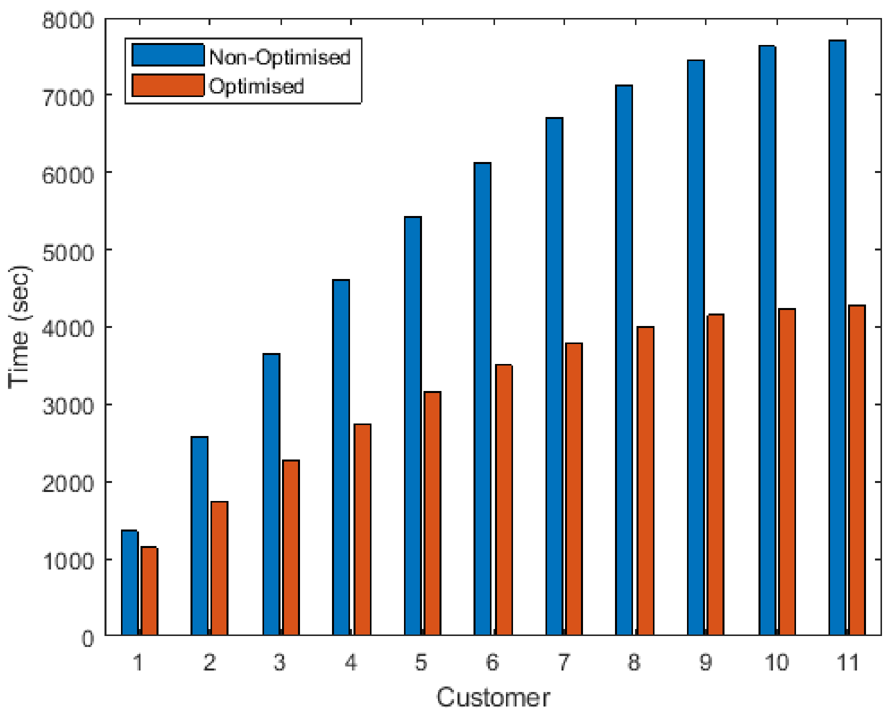


Figure 4. 16 Comparison between the optimized and non-optimized date of delivery for input orders with decreasing customer orders.

As a further test, a sixth instance of inputs were provided to the model. Here, the first customer order was deliberately made quite high and set to vary randomly from then on. The required delivery date was set to be on the same day. This set of inputs and input conditions would test the robustness of the model as well as test if a high initial input has a detrimental effect on the model. The inputs are shown in Table 4.14.

Table 4. 14 Customer requirements table with high initial customer input and same day delivery

Customer Name	Number of 500ml bottles required	Number of 750ml bottles required	Required date and time of delivery
1	200	250	2019/06/16 15:00
2	85	95	2019/06/16 15:00
3	120	120	2019/06/16 15:00
4	100	150	2019/06/16 15:00
5	120	150	2019/06/16 15:00
6	79	69	2019/06/16 15:00
7	25	30	2019/06/16 15:00
8	25	30	2019/06/16 15:00
9	45	50	2019/06/17 15:00
10	100	200	2019/06/17 15:00
11	100	100	2019/06/17 15:00

The results of the optimization are shown in Figure 4.17. As in all five instances previously, the optimization model has ensured that the required date of delivery has been met. In this instance, despite the high initial order and same date of delivery, the results show that the optimized date of delivery is faster than the non-optimized date of delivery.

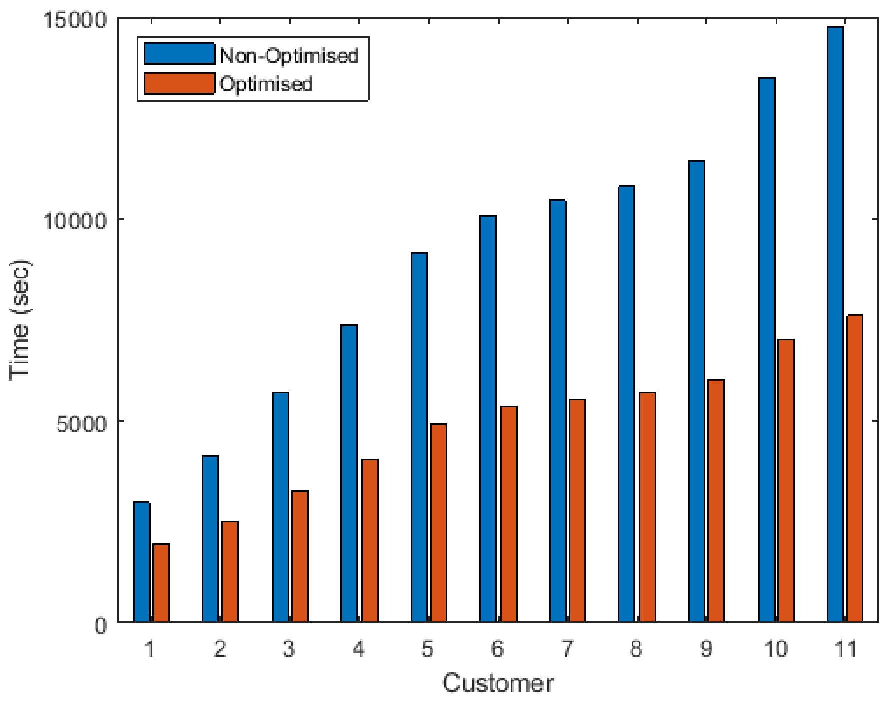


Figure 4. 17 Comparison between the optimized and non-optimized date of delivery for high first customer orders

A similar test with all eleven customer ordering 200 bottles of 500ml and 750ml required to be delivered on the same day was done yielding a similar result. The result is depicted in Figure 4.18.

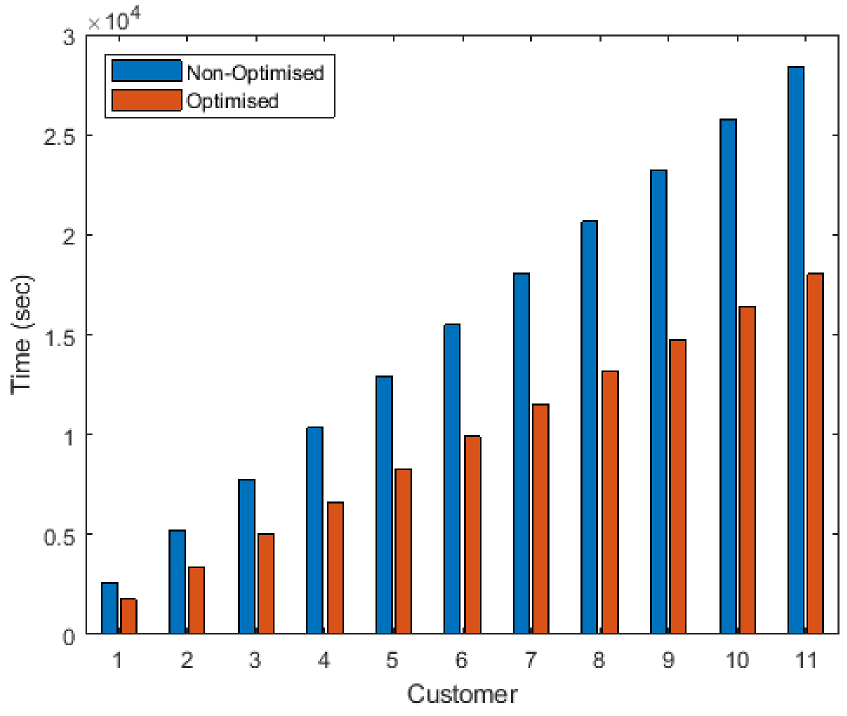


Figure 4. 18 Comparison between the optimized and non-optimized date of delivery for all orders at 200 bottles

A total of twenty special instances were tested (seven of which are analysed in this section) with varying values of customer inputs and required date of delivery. It was seen that the required date of delivery was met on all 20 instances. However, in two instances, depicted in Figure 4.14 and 4.15, the non-optimized delivery date was quicker (Figure 4.14) than or on par (Figure 4.15) with the optimized delivery date for the first customer. It is important to note here that this phenomenon was only visible for the first customer order.

In the twenty special instances tested here, niNe instances were those with order and required delivery dates on the same day. It was seen that in six of the nine instances the non-optimized delivery date was faster than the optimized delivery date for the first customer order. This is similar to the response shown in the third instance depicted in Figure 4.14.

As this would be a demerit on the model, a test was done to check the status of the constraints as these nine instances were input to the model. It was seen that in all six instances, mentioned previously, the level of water in the tank had gone below 25%, which was set as a threshold. This could be the mitigating factor that caused all subsequent customer orders to be faster under optimization.

In order to test if the number of inputs proved to be a hindrance to the optimal operation of the model, four of the twenty special instances had high inputs for both the 500ml and 750ml bottles. It was seen that in all instances the optimized date of delivery was faster that the non-optimized date of delivery. This proved that high inputs were not a factor that affected the functioning of the model.

The remaining instances had random inputs with the required delivery date varying from 30-miuntes to 2 hours apart for each subsequent customer input. It was seen that in all instances except one, the optimized date of delivery was faster than the non-optimized date of delivery. This instance was depicted in Figure 4.15. As mentioned previously, this was evident only for the first customer order. Several other instances were done with no repetitions of the same scenario.

Chapter 5 Research Contributions and Conclusions

5.1 Introduction

This chapter aims to bring to the fore the contributions of the study to the field of knowledge. This is done by revisiting the research goals and objectives of the project, summarizing the achieved results, identifying the future scope of study and finally drawing a conclusion to the research.

5.2 Summary

The first chapter introduced the research project and stated the problem the research aimed to solve. Thereafter, an apt hypothesis was identified and a list of research objectives were stated. The objectives were used to formulate the research methodology of the project. The first chapter was concluded by looking at the layout of the thesis. In the second chapter, a comprehensive literature review was done. This was done by initially focusing on the general problem of assembly line balancing and finally concluded by looking at the limitations of the existing research in the field which necessitated this specific study. The third chapter focused on the research methodology used to solve this problem. This chapter showcased how the water bottling plant, a case study in this project, was modelled, customized and finally made into a make-to-order system using a cloud server. The chapter also detailed the different types of optimization problems with focus on real time optimization and how to formulate optimization problems using MATLAB. The fourth chapter conducts a detailed analysis and depicted the results of the research. This is done by initially focusing on the functioning of the primary model, followed by the customization of the model. Thereafter, a comparison of the non-optimized and optimized operation of the customized make-to-order model is done. The comparison is able to bring to light the problem of positive drift in a real time make-to-order system and how it can be overcome by optimization.

5.3 Research Goals

The main goal of the research was to optimize, in real time, a multi mixed make-to-order assembly line to reduce positive drift. However, this posed a few challenges as existing systems operate on a fixed routine and any disruption might result in a production delay.

Therefore, in order to study this unique problem, firstly, an extensive review of the literature focusing on assembly line balancing needed to be done. Secondly, a model of the case study needed to developed on a software platform so as to conduct experiments. Thirdly, a cloud interface was created to the model. This would allow for real time inputs to be provided to the model. Next, the model was simulated to establish the extend of positive drift in the model. This step paved way for the next step which was to formulate an optimization problem by identifying the plausible constraints and handles. Finally, a comparison is done on the impact of a non-optimized and an optimized handle on positive drift.

The literature review was done in Chapter 2. It focused initially on Assembly Line Balancing and its classification. The chapter then goes to discuss the problems associated with Assembly Line Balancing. In Section 2.3.2, the problem of positive drift, which is the focus of this study, is detailed. This chapter also gives an extensive look into Assembly Line Balancing Techniques and the limitations of existing research. The limitations of the existing research underpin the need for the study expressed in the problem statement.

As mentioned previously, the challenge at hand is too severe to study on an existing system. Therefore, case study needs to be identified, modelled and simulated on a software platform to establish the problem statement. This is done in Chapter 3. Here, a case study of a water bottling plant was chosen. The bottling plant is a prime example of a mixed model assembly line as it produces more than one variant (500ml and 750ml) of the same product. Section 3.3 details how the primary model was developed on Simulink. The inputs to the model will be provided in real time through a cloud interface, therefore the operation of the plant will be stochastic. This is discussed in Section 3.5 This chapter is rounded off by looking at the optimization techniques and how the optimization problem for the plant model was formulated.

The results of the study are collated in a structured manner and described in Chapter 4. Initially, tests are done on the primary model and the results of this test are depicted in Section 4.2. Next the results the customized make-to-order model are portrayed. Thereafter, the results of the non-optimized operation of the model is done. These tests are able to bring to light the problems described in the problem statement and stress the importance for the need of optimization.

Section 4.5 shows the difference the optimized operation brings to the customized make-to-order model.

The chapter is rounded off by looking at veracity and robustness of the developed model. Here a set of twenty instances with diverse values of date and time of delivery and customer inputs were provided to the model. The results show that model was successful in meeting the required date of delivery for all instances. However, in two instances (Third instance and fourth instance), the optimized date and time of delivery was slower than the non-optimized date and time of delivery. This anomaly was only evident for the first customer order. Thereafter for every subsequent customer order, the optimized date and time of delivery was quicker than the non-optimized date and time of delivery.

A closer examination of the constraints, while these two instances were running, showed that pump flow rate for the first customer order in the non-optimized state was very high and started using the water from the tank at a quick rate to complete the order. This resulted in the first order being completed quickly, but all subsequent orders being delayed as it needed the tank to be replenished. The optimized delivery time, meanwhile, factored in the total number of orders and determined a suitable pump flow rate which would ensure that all customer orders be completed at an ideal time while ensuring that the constraints are met.

5.4 Research Contributions

The research project developed a model to optimize a real time multi mixed make-to-order assembly line to reduce positive drift. The following contributions from the study are considered to be novel;

5.4.1 Contributions to existing knowledge

This project studied several journal articles written in the field of assembly line balancing from 1989 [44] (Ghosh and Gagnon) to 2014 [36] (Sivansankaran and Shahabudeen). As part of contributing to new knowledge, a review journal article [20] titled "A Review of the Literature on Assembly Line Balancing Problems, the Methods Used to Meet These Challenges and the Future Scope of Study" was published in 2018. This article was able to capture the updates done in the field of study from 2014 and highlighted the lack of research done on Multi Model Stochastic S-type (MMSS) assembly lines.

As stated in a comparison study in Section 2.4.2, MMSS assembly lines were among the least researched and this study paved way for stating the need for more focused study on Mixed Model Assembly Lines. The next journal article focused on modelling a typical Mixed Model Assembly Line [84] using Simulink and thereafter three journal articles were published on model customization [98], problem formulation [99] and model optimization [100].

A total of FOUR journal articles and TWO conference papers in Engineering were published/in publication at the time of submitting this thesis.

5.4.2 Development of Generic Simulink Model

The inability to test the optimization feature on an operational plant is what necessitated the need for developing general model for Mixed Model Stochastic Assembly Lines. The Simulink model developed in this project could produce two varieties of a product. Through the cloud feature, it could take in numerous inputs in real time and optimization function could take up to three constraints in this study. However, the model is designed such that the product variety and number of constraints could be increased with ease to suit a specific plant requirement.

This would mean that the model can be used as a baseline in future study into Mixed Model Stochastic assembly line balancing and possibly be also used in a virtual commissioning environment. The optimization function used in this study, while being robust and versatile, can be easily changed to suit the needs of a different plant at any point in time. This will allow for tests to be done without structural change, in future, using other optimization tool boxes.

5.4.3 Contributions to Industry 4.0

The fourth Industrial revolution, commonly referred to as Industry 4.0, is already shaping the manufacturing scene for the upcoming future. It incorporates Cyber Physical Systems (CPS), Internet of Things (IoT's) and Internet of Services (IoS's), to name a few, to drive SMART factories. One of the key aspects of SMART factories is Cloud manufacturing. In sensing the importance of research into this new field, this project has integrated a Web enabled input to the model. The Web enabled input acts a gateway to cloud server through which the inputs of the customers are read. This will mean ubiquitous access to the plant.

5.5 Future Work

This thesis has been presented as part of an ongoing research project at the Central University of Technology, Free State, studying the different types of Assembly Line Balancing Problems and how the assembly lines can be optimized to reduce production time, thereby increasing productivity.

The conclusions arrived in this specific study form a platform from which various other studies can be launched. This study specifically focused on mixed model stochastic assembly lines as they were the least researched, however there are many aspects that deem further introspection.

The first of which is to examine if the developed model can be altered so that it can be used as a digital twin for an existing or proposed physical plant. The advantage of such an approach is that, real time optimization can be done prior to plant operation and allow for digital monitoring of plants. This can open research paths into virtual/hybrid commissioning.

Another possible extension of this research is the scope of study into SMART manufacturing with specific focus on CLOUD manufacturing. The protocols that define communication between the CLOUD and the Smart Manufacturing Units (SMU’s) have not been standardized due to the fast pace of development in this field and in the governing field of Industry 4.0. Using this research as a base, study can be done on defining standard protocols.

The model developed in Section 3.5 in this study can also be used create three separate SMU’s. These can be used to investigate the decentralized operation of SMU’s. Decentralized operation is one of the enabling factors of interoperability, a key characteristic of a SMART manufacturing environment. Decentralization allows for direct communication between the SMU’s as opposed to communication between SMU’s through a cloud server which can result in data latency.

5.6 Scientific outcomes

RB Kuriakose and HJ Vermaak, "*A Review of the Literature on Assembly Line Balancing Problems, the Methods Used to Meet These Challenges and the Future Scope of Study*," Advanced Science Letters, vol. 24, No. 11, pp. 8846–8850, 2018.ISSN: 1936-6612, doi:10.1166/asl.2018.12359.

RB Kuriakose and HJ Vermaak, "*Customized Mixed Model Stochastic Assembly LineModelling Using Simulink*," International Journal of Simulation Systems, Science and Technology,vol.20, No. 1, pp. 61–69, 2019. ISSN:1473-804X, doi:10.5013/IJSSST.a.20.S1.06

RB Kuriakose and HJ Vermaak, "*Optimization of a Customized Mixed Model Assembly Line Using MATLAB/Simulink*," Journal of Physics: Conference Series, vol. 1201, no. 1017, 2019. ISSN:1742-6596, doi:10.1088/1742-6596/1201/1/012017.

RB Kuriakose and HJ Vermaak, "*Optimization of a Real Time Web Enabled Mixed Model Stochastic Assembly Line to Reduce Production Time*," Springer CCIS Series, 2019.

RB Kuriakose and HJ Vermaak, "*Developing a Business Plan for an In-University Water Bottling Plant-A Case Study done at Central University of Technology , South Africa*", 2018 International Symposium in Business Management and Social Sciences (BIMSS-2018) 28 – 30 August, 2018 Surabaya, Indonesia

RB Kuriakose and HJ Vermaak, "*Modelling and Simulating a Mixed Model Assembly Line : A Case Study Using a Water Bottling Plant*," 4th International Conference on Computer, Communication and Control Technology (I4CT 2018) 20th – 22nd March, 2018, Krabi, Thailand

References

[1] P. Kaminsky and O. Kaya, "Combined make-to-order make-to-stock supply chains," *IIE Transactions*, vol. 41, no. 1, pp. 103–119, 2009.

[2] H. Lee, "Effective Inventory and Service Management Through Product and Process Redesign," *Operations Research*, vol. 44, pp. 1515–159, 1996.

[3] N. Kumar and D. Mahto, "Assembly Line Balancing: A Review of Developments and Trends in Approach to Industrial Application," *Global Journal of Research in Engineering*, vol. 13, no. 2, p. 23, 2013.

[4] D. Gupta and S. Benjaafar, "Make-to-order , Make-to-stock , or Delay Product Differentiation ? – A Common Framework for Modeling and Analysis," *IIE Transactions*, vol. 36, no. 612, pp. 529–546, 2004.

[5] S. J. Hu *et al.*, "Assembly System Design and Operations for Product Variety," *CIRP Annals - Manufacturing Technology*, vol. 60, no. 2, pp. 715–733, 2011.

[6] J. Um, A. Lyons, H. K. S. Lam, T. C. E. Cheng, and C. Dominguez-pery, "Product Variety Management And Supply Chain Performance : A Capability Perspective On Their Relationships And Competitiveness Implications," *International Jounal of Production Economics*, vol. 187, no. June 2016, pp. 15–26, 2017.

[7] A. Gharehgozl, M. Rabbani, N. Zaerpour, and J. Razmi, "A Comprehensive Decision-Making Structure For Acceptance/Rejection Of Incoming Orders In Make-To-Order Environments," *International Journal of Advanced Manufturing Technology*, vol. 39, no. 9, pp. 1016–1032, 2008.

[8] N. Kriengkorakot and N. Pianthong, "The Assembly Line Balancing Problem :Review Articles," *KKU Engineering Journals*, vol. 34, April, pp. 133–140, 2007.

[9] Z. Yuguang, A. Bo, and Z. Yong, "A PSO Algorithm For Multi-Objective Hull Assembly Line Balancing Using The Stratified Optimization Strategy," *Computers and Industrial Engineering*, vol. 98, pp. 53–62, 2016.

[10] A.-H. Tarek, A. Marwan, A.-A. Omar, and M. Ahmed, "Development Of A Genetic Algorithm For Multi-Objective Assembly Line Balancing Using Multiple Assignment Approach," *International Journal of Advanced Manufacturing Technology*, vol. 77, no. 5, pp. 1419–1432, 2015.

[11] T. Hager, H. Wafik, M. Ahmed, and M. Faouzi, "An Assembly Line Balancing Problem Automotive Cables," *Management and Production Engineering Review*, vol. 6, no. 1, 2015.

[12] C. Becker and A. Scholl, "A Survey On Problems And Methods In Generalized Assembly Line Balancing," *European Journal of Operational Research*, vol. 168, pp. 694–715, 2006.

[13] M. Jamil and N. M. Razali, "Simulation of Assembly Line Balancing in Automotive Component Manufacturing," *Material Science and Engineering*, vol. 114, no. 1, 2016.

[14] J. C. Chen, C. Chen, L. Su, H. Wu, and C. Sun, "Assembly Line Balancing In Garment Industry," *Expert Systems with Applications*, vol. 39, no. 11, pp. 10073–10081, 2012.

[15] A. Allahverdi, T. Cheng, and M. Kovalyov, "A Survey Of Scheduling Problems With Setup Times Or Costs," *European Journal of Operational Research*, vol. 187, no. 3, pp. 985–1032, 2008.

[16] C. Andres, C. Miralles, and R. Pastor, “Balancing And Scheduling Tasks In Assembly Lines With Sequence-Dependent Setup Times,” *European Journal of Operational Research*, vol. 187, no. 3, pp. 1212–1223, 2008.

[17] P. Huang, H. Li, and L. Han, “Order Scheduling Problems In Make-To-Order Manufacturing Systems,” in *IEEE International Conference Mechatronics and Automation*, 2005, vol. 4, no. 1, pp. 2179–2184.

[18] M. Razali, F. Abdul Rashid, and M. Razif Abdullah, “Mathematical Modelling of Mixed-Model Assembly Line Balancing Problem with Resources Constraints,” in *International Engineering Research and Innovation Symposium (IRIS)*, 2016.

[19] P. Cortes, L. Onieva, and J. Guadix, “Optimizing And Simulating The Assembly Line Balancing Problem In A Motorcycle Manufacturing Company: A Case Study,” *International Journal of Production Research*, vol. 48, no. 12, pp. 3637–3656, 2011.

[20] R.B Kuriakose and H.J Vermaak, “A Review of the Literature on Assembly Line Balancing Problems, the Methods Used to Meet These Challenges and the Future Scope of Study,” *Advanced Science Letters*, vol. 24, no. 11, pp. 8846–8850, 2018.

[21] R. Domm, *Michigan Yesterday & Today*, 1st ed. New York: Voyageur Press, 2009.

[22] H. Ford and S. Crowther, *My Life and Work*. Garden City, NY: Doubleday, Page & company, 1922.

[23] S. Pfeiffer, “Robots, Industry 4.0 and Humans, or Why Assembly Work Is More than Routine Work,” *Societies*, vol. 6, no. 16, 2016.

[24] C. B. Frey and M. A. Osborne, The Future Of Employment: How Susceptible are Jobs to Computerisation?, *1st ed. Oxford: Oxford Martin*, 2013.

[25] M. Marcin, “The History of the Assembly Line,” *Crest Capital*, 2018. [Online]. Available: https://www.crestcapital.com/tax/history_of_the_assembly_line. [Accessed: 20-Dec-2018].

[26] R. Pannerselvam and C.San kar, “New Heuristics for Assembly Line balancing Problem,” *International Journal of Management Systems*, vol. 9, no. 1, pp. 25–36, 1993.

[27] B. Timilsina, “Removing Bottleneck From A Manufacturing Unit :A Case Studies to BET-KER OY, Ylivieska-84100, Finland,” *Central Ostrobothnia University Of Applied Sciences*, Ylivieska Unit, 2012.

[28] V. Pachghare and R. S. Dalu, “Assembly Line Balancing – A Review,” *International Journal of Scientific Research*, vol. 3, no. 3, pp. 807–811, 2014.

[29] H. Wang and S. Hu, “Manufacturing Complexity In Assembly Systems With Hybrid Configurations And Its Impact On Throughput,” *CIRP Annals - Manufacturing Technology*, vol. 59, no. 1, pp. 53–56, 2010.

[30] M. Kilbridge and L. Wester, “The Balance Delay Problem,” *Management Sciences*, vol. 8, no. 1, pp. 69–84, 1961.

[31] C. Merengo, F. Nava, and A. Pozetti, “Balancing and sequencing manual mixed-model assembly lines,” *Int. J. Prod. Res.*, vol. 37, no. 1, pp. 2835–2860, 1999.

[32] M. . Grabau and R. . Maurer, “Assembly Line Balancing When Scarp Impacts The Bottom Line,” *Production and Inventory Management Journal*, vol. 39, no. 1, pp. 16–21, 1998.

[33] N. Boysen *et al.*, “A Classification Of Assembly Line Balancing Problems A Classification Of Assembly Line Balancing Problems,” *Jenaer Schriften zur Wirtschaftswiss.*, pp. 1–27, 2006.

[34] A. Scholl, Balancing and Sequencing of Assembly Lines, *2nd ed. Physica-Verlag Heidelberg*, 1999.

[35] N. Kumar and D. Mahto, “Assembly Line Balancing : A Review Of Developments And Trends In Approach To Industrial Application,” *Global Journal of Research in Engineering*, vol. 13, no. 2, 2013.

[36] P. Sivasankaran and P. Shahabudeen, “Literature Review Of Assembly Line Balancing Problems,” *International Journal of Advanced Manufucturing Technology*, vol. 73, no. 9–12, pp. 1665–1694, 2014.

[37] H. Tempelmeier, “Practical Considerations in the Optimization of Flow Production Systems,” *Journal of Production Research*, vol. 41, no. 1, p. 41, 149–170., 2003.

[38] Y. Kara, T. Paksoy, and C. Ter Chang, “Binary Fuzzy Goal Programming Approach To Single Model Straight And U-Shaped Assembly Line Balancing,” *European Journal of Operations Research*, vol. 195, no. 1, pp. 335–347.

[39] P. Sivasankaran and P. Shahabudeen, “Genetic Algorithm for Concurrent Balancing of Mixed-Model Assembly Lines with Original Task Times of Models,” *Intelligent*

Information Management, vol. 5, pp. 84–92, 2013.

[40] G. R. Aase, J. Olson, and M. Schniederjans, “U-shaped assembly line layouts and their impact on labor productivity: An experimental study,” *European Journal of Operations Research*, vol. 156, no. 3, pp. 698–711, 2004.

[41] P. M. Vilarinho and A. N. A. Sofia, “A Two-Stage Heuristic Method For Balancing Mixed-Model Assembly Lines With Parallel Workstations,” *International Journal of Production Research*, vol. 40, no. 6, pp. 1405–1420, 2002.

[42] G. R Esmaeilian, M. Mahmad, S. Sulaiman, and N. Ismail, *Assembly Line and Balancing Assembly Line*. 2018.

[43] P. Sharma, G. Thakar, and R. C. Gupta, “Evaluation Of Multi Criteria Assembly Line Balancing By Mcdm Approaches : A Conceptual Review,” *International Journal of Qualitative Research*, vol. 8, no. 1, pp. 87–106, 2014.

[44] S. Ghosh and R. . Gagnon, “A Comprehensive Literature Review and Analysis of the Design, Balancing and Scheduling of Assembly Systems,” *International Journal of Production Research*, vol. 27, no. 4, pp. 637–670, 1989.

[45] A. Scholl, R. Klein, and W. Domschke, “Pattern Based Vocabulary Building For Effectively Sequencing Mixed Model Assembly Lines.,” *Journal of Heuristics*, vol. 4, no. 1, pp. 359–381, 1998.

[46] T. Urban, “Optimal Balancing Of U-Shaped Assembly Lines,” *Management Sciences*, vol. 44, no. 5, pp. 738–741, 1998.

[47] C. K. Prahalad and V. Ramaswamy, “Co-Creation Experiences: The Next Practice In Value Creation,” *Journal of Interactive Marketting*, vol. 18, no. 3, pp. 5–14, 2000.

[48] T. . Hu, “Parallel Sequencing and Assembly Line Problems,” *Operational Research*, vol. 9, no. 6, pp. 841–848, 1961.

[49] J. . Jackson, “A Computing Procedure for Line Balancing Problem,” *Managememt Sciences*, vol. 2, no. 3, 1956.

[50] M. Klein, “On Assembly Line Balancing,” *Operational Research*, vol. 11, no. 2, 1963.

[51] E. Mansoor and B. Tuvia, “Optimising Balanced Assembly Lines,” *Journal of Industrial Engineering*, vol. 15, no. 2, 1964.

[52] A. Nevins, “Assembly Line Balancing Using Best Bud Search,” *Management Sciences*, vol. 18, no. 9, 1972.

[53] N. Hamta, S. FatemiGhomi, F. Jolai, and U. Bahalke, “Bi-Criteria Assembly Line Balancing By Considering Flexible Operation Times,” *Applied Mathematical Modelling*, vol. 35, no. 12, p. 5592–5608., 2011.

[54] U. Ozcan and B. Toklu, “Multiple-Criteria Decision-Making In Two-Sided Assembly Line Balancing : A Goal Programming And A Fuzzy Goal Programming Models,” *Computers and Operational Research*, vol. 36, pp. 1955–1965, 2009.

[55] S. Topaloglu, L. Salum, and A. Supciller, “Rule-Based Modeling And Constraint Programming Based Solution Of The Assembly Line Balancing Problem,” *Expert System Applications*, vol. 39, no. 3, p. 3484–3493., 2012.

[56] A. Dolgui and O. Battaı, “A Taxonomy Of Line Balancing Problems And Their Solution Approaches,” *International Journal of Production Econonomy*, vol. 142, no. 1, pp. 259–277, 2013.

[57] A. Arcus, “COMSOAL: A Computer Method of Sequencing Operations for Assembly Lines,” *International Journal of Production Research*, vol. 4, no. 4, pp. 259–277, 1966.

[58] E. Dar-EL, “Solving Large Single Model Assembly Line Balancing Problems-A Comparitive Study,” *AIIE Transactions*, vol. 7, no. 3, pp. 302–310, 1975.

[59] W. Helgeson and D. . Birnie, “Assembly Line Balancing Using the Ranked Positional Weight- Technique,” *Journal of Industrial Engineering*, vol. 12, no. 6, pp. 394–398, 1961.

[60] T. Hoffmann, “Assembly Line Balancing with a Precedence Matrix,” *Management Sciences*, vol. 9, no. 4, pp. 551–562, 1963.

[61] E. Mansoor, “Assembly Line Balancing -An Improvement on the Ranked Positional Weight Technique,” *Journal of Industrial Engineering*, vol. 15, no. 2, pp. 73–77, 1964.

[62] M. F. Yegul, K. Agpak, and M. Yavuz, “A New Algorithm For U-Shaped Two-Sided Assembly Line Balancing,” *Canadian Society of Mechanical Engineers*, vol. 34, no. 2, pp. 225–241, 2010.

[63] R. Gamberini, G. Andrea, and R. Bianca, “A New Multi-Objective Heuristic Algorithm For Solving The Stochastic Assembly Line Re- Balancing Problem,” *International Journal of Production Economy*, vol. 102, no. November 2014, pp. 226–243, 2006.

[64] J. Kottas and H. . Lau, “A Cost-Oriented Approach to Stochastic Line Balancing,” *AIIE Transactions.*, vol. 5, pp. 164–171, 1973.

[65] J. Rubinovitz and G. Levitin, “Genetic Algorithm For Assembly Line Balancing,” *International Journal of Production Economics*, vol. 41, pp. 343–354, 1995.

[66] Y. Kim, “Genetic Algorithms For Assembly Line Balancing With Various Objectives,” *Computers and Industrial Engineering*, vol. 30, no. 3, pp. 397–409, 1995.

[67] J. Gao, L. Sun, L. Wang, and M. Gen, “An Efficient Approach For Type Ii Robotic Assembly Line Balancing Problems,” *Computers and Industrial Engineering*, vol. 56, pp. 1065–1080, 2009.

[68] M. Gen, Y. Tsujimura, and Y. Li, “Fuzzy Assembly Line Balancing Using Genetic Algorithms,” *Computers and Industrial Engineering*, vol. 31, no. 3–4, pp. 631–634, 1996.

[69] P. Zacharia and A. Nearchou, “A Meta-Heuristic Algorithm For The Fuzzy Assembly Line Balancing Type-E Problem,” *Computers and Operational Research*, vol. 40, pp. 3033–3044, 2013.

[70] A. Baykasoglu and L. Ozbakir, “Stochastic U-Line Balancing Using Genetic Algorithms,” *International Journal of Advanced Manufcturing*, vol. 32, pp. 139–147, 2007.

[71] A. Haq and J. Jayaprakash, “A Hybrid Genetic Algorithm Approach To Mixed- Model Assembly Line Balancing,” *International Journal of Advanced Manufacturing*, vol. 28, no. March, pp. 337–341, 2006.

[72] P. Chutima and P. Olanviwatchai, “Mixed-Model U-Shaped Assembly Line Balancing Problems with Coincidence Memetic Algorithm,” *Journal of Software Engineering and Applications*, vol. 3, pp. 347–363, 2010.

[73] S. D. Lapierre, A. Ruiz, and P. Soriano, “Balancing Assembly Lines With Tabu Search,” *European Journal of Operation Research*, vol. 168, pp. 826–837, 2006.

[74] E. Erel, Y. Gocgun, and I. Sabuncuoglu, “Mixed-Model Assembly Line Sequencing

Using Beam Search," *International Journal of Production Research*, vol. 45, pp. 5265–5284, 2007.

[75] U. Ozcan and B. Toklu, "A Tabu Search Algorithm For Two-Sided Assembly Line Balancing," *International Journal of Advanced Manufacturing Technology*, vol. 43, no. 822, 2009.

[76] P. McMullen and P. Tarasewich, "Using Ant Techniques to Solve the Assembly Line Balancing Problem," *IIE Transactions*, vol. 35, no. 7, pp. 605–617, 2003.

[77] P. McMullen and G. Frazier, "Using Simulated Annealing to Solve a Multi-objective Assembly Line Balancing Problem With Parallel Workstations," *International Journal of Production Reserach*, vol. 36, no. 10, 2001.

[78] W. Xu and T. Xiao, "Mixed Model Assembly Line Balancing Problem With Fuzzy Operation Times And Drifting Operations," in *Proceedings of the 2008 Winter Simulation Conference*, 2008, pp. 1752–1760.

[79] S. Matanachai and C. A. Yano, "Balancing Mixed-Model Assembly Lines To Reduce Work Overload," *IIE Transactions*, vol. 33, no. 1, pp. 29–42, 2001.

[80] P. Y. Tambe, "Balancing Mixed-Model Assembly Line To Reduce Work Overload In A Multi-Level Production System," Louisiana State University, 2006.

[81] "Central University of Technology." [Online]. Available: http://www.cut.ac.za/. [Accessed: 01-May-2015].

[82] F. Maile, H. Vermaak, and N. Luwes, "Documentation," Bloemfontein, 2017.

[83] R.B Kuriakose and H.J Vermaak, "Developing a Business Plan for an In-University Water Bottling Plant-A Case Study done at Central University of Technology , South Africa," *J. Adv. Manuf. Technol.*, vol. Awaiting p, 2019.

[84] R. B. Kuriakose and H. J. Vermaak, "Modelling and Simulating a Mixed Model Assembly Line : A Case Study Using a Water Bottling Plant," *J. Telecommun. Electron. Comput. Eng.*, vol. X, no. X, pp. 1–5.

[85] R. Kuriakose and H. Vermaak, "Modelling a Customized Mixed Model Assembly Line: A Case Study Using a Water Bottling Plant," *Telecommun. Comput. Electron. Control*, vol. Awaiting p, 2018.

[86] N. Kumar, “Optimization:Introduction and Basic Concepts,” *Optimization Methods*. [Online]. Available: https://slideplayer.com/slide/7523540/. [Accessed: 28-Dec-2018].

[87] N. Kumar, “Optimization problem and Model formulation,” *Optimization Methods*. [Online]. Available: http://msulaiman.org/onewebmedia/Lecture 2 mphil.pdf. [Accessed: 28-Dec-2018].

[88] G. Venter, “Review of Optimization Techniques,” in *Encyclopedia of Aerospace Engineering*, vol. 8, no. 1, B. Richard and S. Wei, Eds. New York: John Wiley & Sons, 2010, pp. 1–12.

[89] Galperin.EA, “Problem-Method Classification in Optimizatin and Control,” *Computers and Mathematical Applications*, vol. 21, no. 6, pp. 1–6, 1991.

[90] A. Astolfi, “Optimization:An introduction,” *Imperial College London*, 2006. [Online]. Available: http://www3.imperial.ac.uk/pls/portallive/docs/1/7288263.PDF.

[91] N. Kumar, “Introduction and Basic Concepts -Classification of Optimization Problems,” 2006.

[92] D. Bonvin, “Preface to ‘Real-Time Optimization,’” *Processes*, vol. Special e, pp. 1–5, 2017.

[93] T. F. Edgar and J. Hahn, "*Process Automation"*, Handbook of Automation, pp. 529-543, SSpringer, Newyork, 2009.

[94] J. Hahn and T. F. Edgar, *Kirk-Othmer Encyclopedia of Chemical Technology-Process Control*. London: John Wiley & Sons, 2014.

[95] S. Shokri, R. Hayati, M. Mavast, M. Ayazi, and H. Ganji, “Real Time Optimization as a Tool for Increasing Petroleum Refineries Profits,” *Petroleum and Coal*, vol. 51, no. 2, pp. 110–114, 2009.

[96] T. Edgar, D. Himmelblau, and L. Ladson, *Optimization of Chemical Processes*, 2nd ed. New York: McGraw-Hill Higher Education, 2001.

[97] MATLAB, “MATLAB Optimization ToolBox:User’s Guide,” *MATLAB Documentation*, 2018. [Online]. Available: https://www.mathworks.com/help/pdf_doc/optim/optim_tb.pdf. [Accessed: 02-Jan-2019].

[98] R.B Kuriakose and H.J Vermaak, “Customized Mixed Model Stochastic Assembly Line

Modelling Using Simulink," *International Jounal of Simulation Systems Sciences and Technology*, vol. 20, no. 1, pp. 61–69, 2019.

[99] R.B Kuriakose and H.J Vermaak, "Optimization of a Customized Mixed Model Assembly Line Using MATLAB/Simulink," *Journal of Physics: Conference Series, vol. 1201, no. 1017, 2019. ISSN:1742-6596, doi:10.1088/1742-6596/1201/1/012017*

[100] R.B Kuriakose and H.J Vermaak, "Optimization of a Real Time Web Enabled Mixed Model Stochastic Assembly Line to Reduce Production Time," *Springer CCIS Ser.*, 2019.