Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 217 (2023) 630–639

Procedia Computer Science

www.elsevier.com/locate/procedia

4th International Conference on Industry 4.0 and Smart Manufacturing

Using Simulation Optimization to Improve the Performance of an Automated Manufacturing Line

Patrick Ruane[a]*. Patrick Walsh[b]. John Cosgrove[c]

[a] *Johnson & Johnson Vision Care and Technological University of the Shannon, Limerick, Ireland*

[b] *Technological University of the Shannon, Limerick, Ireland*

[c] *Technological University of the Shannon, Limerick, Ireland*

Abstract

As manufacturing capital equipment is expensive, it is necessary that the equipment once in operation is reliable and delivers to the business plan targets. Simulation along with an optimization system is an invaluable tool to confirm that an automated manufacturing line can produce to the required business objectives before and after it goes into operation. Simulation in manufacturing is often applied in situations where conducting experiments on a real system is very difficult often because of cost or the time to carry out the experiment is too long. Optimization is the organized search for such designs and operating modes to find the best available solution from a set of feasible solutions. It determines the set of actions or elements that must be implemented to achieve an optimized manufacturing line. As a result of being able to concurrently simulate and optimize equipment processes, the understanding of how the actual production system will perform under varying conditions is achieved. Implementing the actual changes to equipment to improve reliability can be both time consuming and expensive. Simulation in conjunction with optimization can be used to verify these improvements before the equipment is modified. This study has adopted an open-source simulation tool (JaamSim) to develop a digital model of an automated tray loader manufacturing system in the Johnson & Johnson Vision Care (JJVC) manufacturing facility. This paper demonstrates how this digital model was integrated with SimWrapper optimization and how both tools can be used for the optimization and development of an automated manufacturing line in the medical devices industry.

* Corresponding author.
E-mail address: patrick.ruane@tus.ie

Keywords: Simulation; Optimization; Digital Model; Digitalization; JaamSim; SimWrapper

10.1016/j.procs.2022.12.259

1. Introduction

Digitalization in manufacturing is the conversion of information into digital format, the integration of this digital data and technologies into the manufacturing process and the use of those technologies (eg: simulation, optimization) to change a business model to provide new revenue and value-producing opportunities. Digitalization may be seen as the increased generation, analysis, and use of data to improve the efficiency of the overall manufacturing system. Digital manufacturing technologies, such as simulation models, have been considered an essential part of the continuous effort towards improving the performance of automated manufacturing equipment and processes. These technologies form the basis of an overall digital manufacturing system that enables the optimization of a manufacturing line during the line design stage or when the line is put into operation. The use of this technology gives a deeper understanding of what can occur on the manufacturing line when it is running. A simulation model when combined with optimization engine, can be used to identify problems before they occur and aid in the selection of optimum parameters to run the line before it is fully designed or built. Digital model and optimization technologies supports other Industry 4.0 technologies such as predictive maintenance, OEE improvement, waste reduction, improve batch changeover times and to improve product quality [1]. In addition, having a digital model enables virtual line analysis, removing the physical restraints of expert engineers having to be on your location [2]. Optimization seeks the maximum or minimum value of an objective function corresponding to variables defined in a feasible range or space. More generally, optimization is the search of the set of variables that produces the best values of one or more objective functions while complying with multiple constraints. The purpose of optimization has been described as objective function, loss function, or cost function for minimization and utility function or fitness function for maximization. [3] [4]. In this paper, it will be referred to as objective function. Simulation optimization (SO) refers to the optimization of an objective function subject to constraints, both of which can be evaluated through a stochastic simulation/digital model [5]. The term simulation optimization (SO) is an overall term for techniques used to optimize stochastic simulations. Simulation optimization involves the search for those specific settings of the input parameters to a stochastic simulation such that a target objective, which is a function of the simulation output, is either maximized or minimized [5]. Simulation techniques allow for modelling and artificially reproducing complex systems using stochastic distributions [6]. Complex simulation models may require long development times and difficult verification and validation processes and finally, simulation is not an optimization tool on its own [7]. According to [7], large Combinatorial Optimization Problems (COPs) require the use of metaheuristics to conduct an efficient search, where he proposes to combine simulation with metaheuristics to form a new class of optimization algorithms called ‘simheuristics’. These algorithms integrate simulation (in any of its variants) into a metaheuristic-driven framework to solve complex stochastic COPs. A metaheuristic is a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms [8]. It was desired that the following three (3) significant issues were considered and if they could be answered by this study:

- Optimized Time to delivery: The need for a way to reduce the lead-time for the equipment vendors to design, build and deliver manufacturing lines to JJVC.
- Line Capacity Prediction: The need for a system that could predict line throughput for business management purposes.
- Line Continuous Improvement: The need to create a system for the manufacturing engineer to enable them to test line improvements before implementation on an operational line.

A significant element of this research is to show how the development of a digital simulation model could answer the above three issues while also contributing to the optimization of the manufacturing line design. The simulation package (JaamSim) used in this industrial case study does not have the capability to perform optimization analysis [9]. We propose to integrate the SimWrapper optimization engine developed by OptTek Systems Inc., [10] with the JaamSim digital model, enabling the optimization of the industrial tray loader digital model/system. This paper reviews the development of the industrial case digital model (Tray Loader System), validation of this model and the integration of a black box optimization engine (SimWrapper) with the digital model. Finally, results obtained from the experiments are given and discussed.

2. Development of the Tray Loader Digital Model

2.1 *Overview of the Tray Loader Digital Model:*

A digital model of an industrial system (Fig 1) known as a Tray Loading System was developed using JaamSim software.

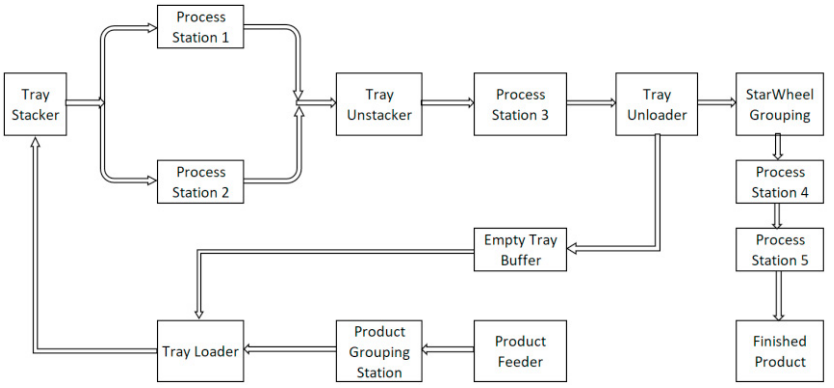

Fig 1: Automated Tray Loading System Industrial Case

This system consists of individual product (p) that arrives from an upstream line to a product feeder at defined arrival times. These are then grouped into multiples of 10. The group of products are then loaded into empty plastic trays that can hold up to 660 parts. Once filled the plastic tray moves at a defined cycle time to a tray stacker. The tray stacker accumulates the filled trays into groups of 30. This group of 30 trays then undergoes a batch process in either Process station 1 or 2 under defined conditions. Upon completion of this batch process, the trays of product leave Process Station 1 or 2, where a tray unstacking operation takes place. Each individual tray of product undergoes a further process step (Process Station 3), again under defined conditions. Once a tray is finished at Process Station 3, the product is removed from the tray at the Tray Unloading station and is then passed to the Star Wheel grouping station, where the product is now grouped into batches of 30. These groups are then passed to Process Station 4 and 5 for the final finishing process. The empty trays from the tray unloading station, are returned to the empty tray buffer and finally back to the tray loader operation, to repeat the overall process. The digital model developed, will simulate this whole operation, considering the following 5 points:

- Entities (units of Product) per arrival.
- Service times for process stations, travel times for conveyors
- Probability distributions for reliability and repair of stations.
- Conditions for process stations to process and pass product to the next station.
- Queue size and location.

2.2 *Verification of the Tray Loader Digital Model:*

A detailed verification process was undertaken on the Tray Loader digital model following the Logical/mathematical verification and program/code verification steps outlined by [11]. All the Tray Loader Objects, Service Times, Steps, Thresholds, Maintenance conditions and Threshold condition logic were all verified and confirmed to be correct to how the actual line operates. A detailed verification checklist was completed on the Tray Loader digital model. As part of the digital model verification process it was important to verify that the product flow into and out of the various simulation objects (as seen from the JaamSim GUI) are identical to what occurs on the tray loader line. This verification process allowed any additions or changes to the simulation logic to be corrected, verified, and visualized immediately. It was through the ongoing and iterative model verification and the testing process during model development, that a realistic model of the actual dynamic interactions was developed and fine-tuned. During this phase of model verification, the weak points of the system were discovered and corrected. It is extremely advantageous to find these early-stage simulation bugs, thus allowing a well-tested and robust system to be developed.

2.3 Validation of the Tray Loader Digital Model

The approach taken for developing the Tray Loader digital model followed the steps described by [12]. This approach also outlines the steps required to validate the programmed digital model. The model is run using the standard basic settings from the actual tray loader system. The simulation model output data for the system was compared with the comparable output data collected from the actual system. This is called results validation. If the results are consistent with how the system should operate, then the simulation model is said to have face validity. Sensitivity analyses is performed on the programmed model to see which factors have the greatest impact on the performance measures and, thus, must be modelled carefully [12]. According to [13], validation is concerned with determining whether the conceptual digital model (as opposed to the computer program) is an accurate representation of the system under study. [13] outlines the following three (3) steps to validate a simulation model.

- Obtaining real-world data from the actual system.
- Tests for comparing simulated and real data (namely graphical, Schruben-Turing or t tests).
- Sensitivity analysis (using statistical design of experiments with associated regression analysis).

The above approach was used to validate the Tray Loader digital model. Actual Tray Loader system data was collected from the historian database for all the relevant process stations used in the digital model. The data collected included input feed rate, yield, throughput and uptime per minute for each process station. Excel macros were then developed to calculate the equipment reliability metrics namely: Mean Time Between Failures (MTBF) and Mean Time to Repair (MTTR) for each of the process stations using the uptime/minute data. The Input feed rate, yield, output data and the MTBF/MTTR for each process station was analysed, outliers removed, and distributions determined along with the distribution parameters. Minitab is used to analyse all the data obtained. Minitab is a statistical analysis software that assists in the analysis of data collected from any process and provides a simple, effective way to input the data, manipulate that data and statistically analyse it. The results from the simulation run were then compared with data from the actual Tray Loader system over 3 months. The mean(μ) and standard deviation(σ) from the simulation results and actual line data are statistically compared to each other to confirm that the simulation model is a true representation of the actual Tray Loader system. Using the 2-sample t test, the mean (μ) from both populations was not significantly different to each other with a P value = 0.609 (P = 0.05 => 95% confidence that means are similar). Based on the analysis of the simulation model and the high level of accuracy with the empirical data gathered from the actual production line, the approach gives a high degree of confidence that the Tray Loader digital model is valid, accurately represents the real physical and operational production environment and provides a solid basis for the further development of the digital model for this industry use-case.

3. Integration of Optimization Engine

3.1. Black Box Optimization and SimWrapper

As described in [14], black box optimizers have a long tradition in the field of operations research. These procedures treat the objective function evaluation as a black box and therefore do not take advantage of the problem's specific structure. Black box optimizers have also been referred to as context-independent procedures, but no solver is totally independent from the context of the problem to be optimized. Some black box optimization engines are developed within a spectrum that ranges from almost no dependence on context to total dependence on context.

A general-purpose commercial optimization software known as SimWrapper® operates by treating the objective function evaluation as a black box. However, SimWrapper® is not totally unaware of the context because the selection of the solution representation gives some information to the optimization engine. SimWrapper allows users to represent solutions as a mixture of continuous, discrete, integer, binary, permutation, and/or other specialized variables. The solution representation gives SimWrapper some information about the problem context and therefore the solver context-independence changes with each application because the amount of information that it receives varies. The SimWrapper® software chooses solvers based on the characteristics of the optimization model: pure or mixed, constrained, or unconstrained and deterministic or stochastic. SimWrapper® main optimization engine is based on the scatter search methodology coupled with tabu search strategies to obtain high quality solutions to problems defined in complex settings. The optimization technologies within SimWrapper® include a list of search procedures

and a separate list of solution generation methods. While the main optimization procedure is based on scatter search, the following seven (7) technologies are also included in SimWrapper to complement the default search mechanisms.

- Design of Experiments (DOE)
- Cross Entropy (CE)
- Genetic Algorithms (GA)
- Particle Swarm Optimization (PSO)
- Simultaneous Perturbation Stochastic Approximation (SPSA)
- Linear and Mixed Integer Programming (LMIP)
- Complete Enumeration

SimWrapper includes a variety of procedures to combine solutions and uses a reactive strategy to select a particular combination for a given subset of solutions. The reactive strategy is such that a success score is kept for each combination method in the catalogue. The score is used to adjust the probability that a particular combination method is selected, where the probability increases with the score value [14] [15].

3.2. Integration of JaamSim Tray Loader Digital Model with SimWrapper

A closed loop digital model and optimization engine was developed. SimWrapper® was integrated with the JaamSim Tray Loader digital model with the following four (4) elements being executed automatically until an optimized solution is obtained:

- Digital Model inputs parameters updated.
- Simulation runs executed and monitored.
- Digital Model outputs collected.
- Optimization analysis completed and new parameter settings recommended.

Python code was developed that integrates the excel input configuration files with both the JaamSim Tray Loader digital model and SimWrapper optimization engine. The structure and code that was written to integrate SimWrapper and the tray loader digital model to form an optimization system followed all the code generation best practices highlighted by [16]. The overall system architecture is shown in Fig 2. This architecture gives a high-level overview of how the optimization system was developed with the main optimization system being controlled by the module called *invoke_Simw* (see purple box in Fig 2). This main function module (*invoke_simw*), then calls other blocks (3 red boxes in Fig 2) forming the main spine of the Tray Loader Optimization system. All the function modules are written using the python programming language. The four (4) pillars of the system include:

- Main controlling function module called *invoke_simw*
- Input Data Pre-processing function block that calls several sub function modules.
- Overall SimWrapper and JaamSim Optimization Loop block calling several sub function modules.
- Output data file post processing block calling several sub function modules.

Tray Loader digital model parameters are passed to the '*invoke_simw*' function. The *invoke_simw* function (See Fig 2, Purple Box), then schedules the calling of all the various functions and methods required to execute all the tasks in the three (3) red boxes. When all input data pre-processing is completed (Fig 2, Red Box 3), the SimWrapper and JaamSim Optimization loop (Fig 2, Red Box 4) is activated where simulation runs are completed using the tray loader digital model.

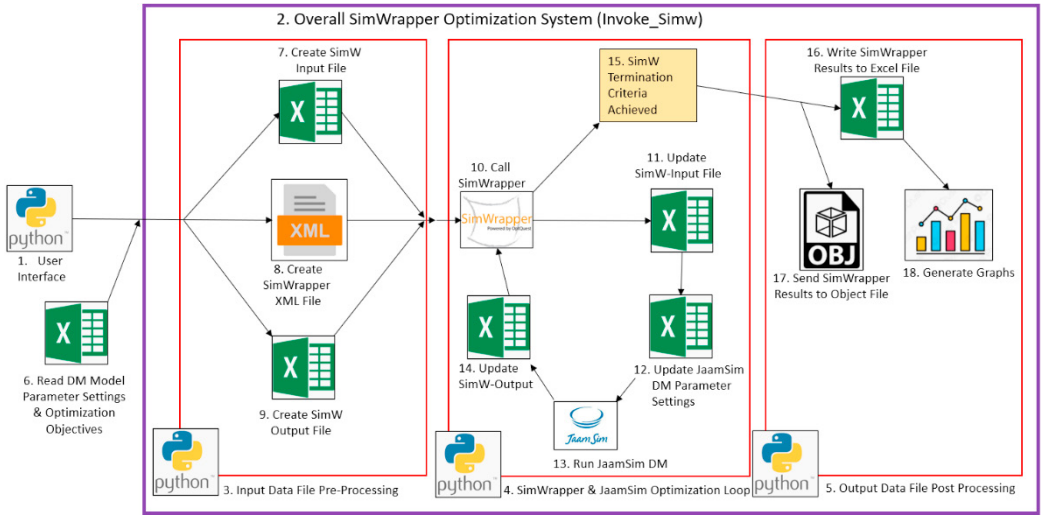

Fig 2: Overall Optimization System Program Architecture and Data Structure

This process is repeated until the termination criteria (Fig 2, block 15) is achieved thus producing an optimal solution. Once the optimization process is terminated, then results from the study are tabulated into excel files, graphs and a SimWrapper object file (Fig 2, Red Box 5). SimWrapper is configured to provide an intuitive, full-featured user interface that will allow the manufacturing engineer to set up, execute, and analyse the results of a simulation study. The SimWrapper tool uses advanced metaheuristic search methods as mentioned in Section 3.1 to drive iterative simulation runs with different simulation input combinations. After the completion of the search, SimWrapper automatically applies different statistical and data mining techniques to provide insight into the influence of the variables on the objectives and to identify good and bad regions of the search space. The optimization portion of SimWrapper is provided by the OptQuest® engine which provides mathematical programming and metaheuristic search libraries. The analysis portion of SimWrapper is provided OptAnalysis® which provides data mining and statistical libraries. Visual plots of optimization progress and simulation response surfaces are provided. SimWrapper uses a statistical and data analysis toolkit to determine the simulation inputs that have the most influence on simulation outputs. This toolkit also helps the engineer to identify meaningful insights from an expansive set of simulation runs. The overall optimization system developed by this study allows the user to specify the objectives to be optimized from an excel file. See Table 1 shows an example of two (2) objectives to be minimized along with two (2) objectives to be maximized (station throughputs). This file is used to configure the optimization problem along with the associated objectives to be either maximised or minimised. Another excel file is set-up to store all the entities/workstations names along with their associated base parameter values, see Table 2 for a sample of some configuration settings for the Tray Loader model.

Table 1. Optimization Objectives.

Processes	Objective
Max Trays Used	Minimize
P_Feeder	Maximize
Process4	Maximize
P_Feeder Util	Minimize

Table 2. Optimization Objectives.

Entity_Name	Parameter	Value	Optim_Space (%)
P_Feeder	InterArrivalTime	0.90 sec	-10 to 10
P_Feeder	Downtime Duration (MTTR)	2.00 Min	-10 to 10
Tray_Pack	Downtime Duration (MTTR)	2.00 Min	-10 to 10
Process4	ServiceTime	2.30 sec	-10 to 10
Process5	ServiceTime	2.30 sec	-10 to 10
P_Feeder_Yield	Weibull Location	0.5573	-10 to 10
Empty_Tray_Stacker	Capacity	90 Units	-15 to 15

The settings in the Optim_Space column are used by SimWrapper. As an example, referring to Table 2, JaamSim is configured with an entity generator called *P_Feeder*. The base *InterArrivalTime* for this generator is 0.90sec. When performing an optimization analysis, the *InterArrivalTime* for this entity can be changed within a space of 0.90 sec ±10% in increments of 1%. Likewise, the Tray Loader JaamSim model uses a resource called *Empty_Tray_Stacker*, with a base setting of 90 units. The optimization space for this parameter is 90 ± 15% in increments of 1%. The user can select which parameters, the base setting for that parameter and if required the optimization space for that parameter to be used by SimWrapper. Reviewing the optimization space for the 7 factors in Table 2, there is in excess of 2.6 billion combinations of different factor settings that the Tray Loader line can be operated to. It is impossible to run all of those combinations using the Tray Loader digital model, hence the need to use optimization approaches to determine a particular setting for each of the 7 factors that results in an optimum solution to the required objective(s). Output results from each simulation run is then analyzed by SimWrapper and any associated changes to the digital model input parameters based on the requirements of the objective function are then made. This process is repeated until an optimal solution is obtained and the optimization process is then terminated. Once SimWrapper optimization has terminated the program returns to the calling function 'invoke_simw'. At this point the function 'Output Data File Post Processing' Fig 2 is called. This block of code prepares the results from the optimization study for review and graphing. The data is also saved to a csv file to allow the user to further analyze the data with statistical packages (eg: Minitab ©) to support any decisions in relation to possible design changes to the tray loading system.

4. Results from the Tray Loader Digital Model and SimWrapper Optimization Engine

All simulation/optimization runs were completed using a HP ZBook Firefly 15 G7 2Z4F7UC laptop running an Intel(R) Core(TM) i7-10810U CPU @ 1.61 GHz processor and 64GB of RAM. The single objective optimization run (maximize *P_Feeder* output) was executed 10 times as recommended by [17] [18]. The results of the 10-run experiment is given in Table 3. As can be seen from Table 3, the *P_Feeder* Mean, Max, Min and Standard Deviation is calculated across the best 100 solutions for each run using the SimWrapper optimization. The average *P_Feeder* (Max) across the 10 runs using SimWrapper was 461,517. The overall *P_Feeder* maximum output across the 10 runs using SimWrapper optimization was 462,664 obtained on run #8. Run #2 was analysed in additional detail, as the results of this particular run produced results that were close to the overall average of the 10 runs completed. Fig 3 shows the results of Run #2 experiment, which consisted of 800 simulation runs/iterations. The *P_Feeder* maximum of 461,631 units was achieved on iteration 313 of the 800 iterations. The blue line represents where the optimum solution sits relative to all other solutions. SimWrapper provides some statistical tools to analyse the results obtained from an optimization run.

Table 3: SimWrapper Single Objective Optimization Results

Run Number	# of Iterations	Execution Time (Mins)	P_Feeder Mean	P_Feeder Max	P_Feeder Min	P_Feeder STD
1	800	102	452,089	460,854	448,547	2,760
2	800	101	454,250	461,631	450,689	2,513
3	800	105	452,999	461,917	449,352	3,059
4	800	96	453,467	462,391	450,491	2,584
5	800	102	451,088	460,344	447,158	2,859
6	800	102	453,426	461,949	449,932	2,696
7	800	110	451,805	460,750	448,458	2,806
8	800	105	453,473	462,664	450,312	2,741
9	800	109	452,905	460,002	450,101	2,187
10	800	107	453,537	462,664	450,685	2,477
Average	800	104	452,904	461,517	449,573	2,668

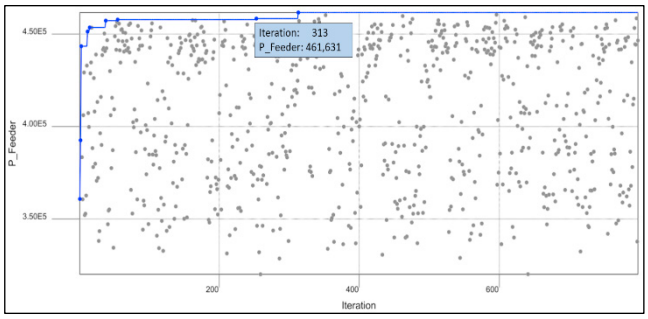

Fig 3: SimWrapper Optimization of P_Feeder output

Variable sensitivity analysis (Fig 5) is completed for run #2 using SimWrapper. The variable sensitivity's view provides an insight into how the input parameter values that were varied during the optimization run relate to and influence the objective or output selected (*P_Feeder* output). As, can be seen from Fig 4, three (3) conclusions can be drawn from the data obtained for this run, these being:

- *P_Feeder InterArrivalTime* has the greatest influence on *P_Feeder* output with an influence scoring of 60 and an R^2 value of 0.8275. Increasing/reducing the *P_Feeder InterArrivalTime* by 1% has the effect of reducing/increasing the *P_Feeder* output by 89,300 units respectively.
- *P_Feeder yield* is 2nd in the influence rank with a score of 43 and an R^2 value of 0.5743 Increasing/reducing the yield by 1%, has the effect of increasing/reducing P_Feeder output by 67,195 units respectively.
- Tray_Pack DownTime duration and Process5_ServiceTime have the least effect on P_Feeder output, due to the low R^2 value and lower value of variable effects than the other input parameters.

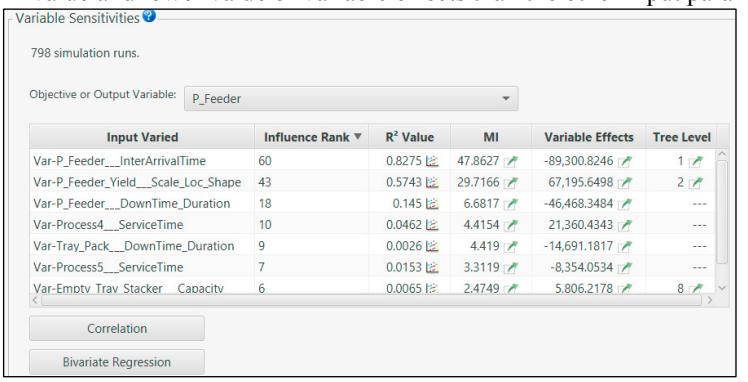

Fig 4: SimWrapper Variable Sensitivity Analysis

The solutions developed using the Tray Loader JaamSim digital model along with SimWrapper optimization engine is shown in Table 4.

Table 4: Tray Loader Optimized Digital Model Parameters

Digital Model Used	Optimization Engine Type	P_Feeder (Max)	P_Feeder_ Inter Arrival Time (Sec)	P_Feeder_ Yield (Location)	P_Feeder_ Down Time_ Duration (Min)	Process4 _Service Time (Sec)	Tray_Pack_ Down Time_ Duration (Min)	Process 5_Service Time (Sec)	Empty_Tray_Stacker_ Capacity Count
Tray Loader JaamSim	SimWrapper	462,664	81	0.613	1.80	2.14	2.18	2.53	94

A two (2) objective optimization problem (maximize the *P_Feeder* and minimize the Empty Tray Buffer capacity) was designed and tested using the tray loader digital model and SimWrapper. As with the single objective optimization problem, the same Tray Loader simulation model, model parameters and optimization parameter space was used for this study (See Table 2), with results given in Table 5.

Table 5: Two Objective Optimization problem of Tray Loader System using SimWrapper

Run Number	# of Iterations	Execution Time (Mins)	P_Feeder Min	P_Feeder Max	Avg_Trays _Used (Min)	Avg_Trays_ Used (Max)
1	1,000	157	314,599	461,993	64	101
2	999	128	310,811	458,465	64	101
3	995	132	311,233	460,083	63	101
4	997	136	312,670	460,221	63	101
5	1,000	134	315,313	461,571	64	101
6	996	151	314,792	461,808	64	102
7	995	137	318,526	459,497	64	101
8	995	137	317,866	457,833	64	101
9	995	123	314,704	458,028	64	101
10	995	128	313,646	460,453	63	101
Average	997	136	314,416	459,995	64	101

A pareto front is a set of nondominated solutions, being chosen as optimal, if no objective can be improved without sacrificing at least one other objective [19]. The pareto front is an excellent visualization to show the interaction of

each objective has on the other. A pareto front (*Empty Tray Buffer* Capacity vs *P_Feeder* Output/Shift) was generated using all the data gathered from the 2 objective SimWrapper Optimization runs. See Fig 5 for the pareto front.

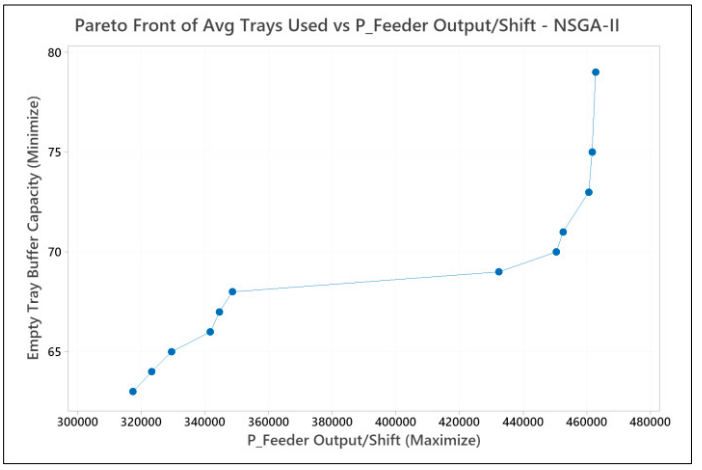

Fig 5: Tray Loader 2 Objective Optimization Pareto Front.

As can be seen from Fig 5, a range of optimum solutions can be obtained (blue line) where a trade-off is required between the *P_Feeder* output and *Empty_Tray_Capacity*. As our primary goal is to maximize *P_Feeder* output and then minimize *Empty_Tray_Capacity*, then the optimum solution is where the *Empty_Tray_Capacity* is approx. 75 trays, thus producing a stable *P_Feeder* output of ~ 458,000.
Increasing the Empty Tray buffer beyond 75 trays, has little impact on the *P_Feeder* output/shift. Since the optimization problem is to maximize *P_Feeder* output and minimize Empty Tray Buffer, the factor settings providing the solution of 75 trays and P_Feeder output of 458,000 is used.

5. Conclusion

This study combined a simulation/digital model developed using JaamSim with an optimization engine (SimWrapper) to provide the design/manufacturing engineer with a valuable digitalization template/method to optimize the design of automated equipment with the aim of improving line performance. The digital model was derived from a real-world automated manufacturing line and validated against actual line performance data. The model was then integrated with a SimWrapper black box optimization system using python code. This research has demonstrated how the development of digital model can be validated and subsequently used as part of an optimization system which is then used for the study of equipment design, maintenance and reliability of an automated production line in the medical devices industry. We have also shown that the use of a digital model/simulation model with an optimization engine can assist the designer to selecting the appropriate factors and associated settings to achieve optimum line performance early in the critical design phase of the project. This research uses simulation optimization which is a family of stochastic optimization techniques including genetic algorithms to support the decision-making process during the early design stage of designing a new automated manufacturing line. Further research in this field is being done by the authors to develop and integrate a specific genetic algorithm optimization system rather than a black box optimization (SimWrapper) engine. This new optimization engine allows the user to tailor the system to their specific application with the aim of developing and finding better optimum solutions for both the design of the manufacturing line and/or parameter settings to run the line at.

References

[1] G. Shao, C. Laroque, S. Jain, L. Hay Lee, P. Lendermann and O. Rose, “Digital Twin for Smart Manufacturing: The Simulation Aspect,” 2019.

[2] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei and L. Wang, “Enabling Technologies and Tools for Digital

Twin," *Journal of Manufacturing Systems,* vol. 58, pp. 3-21, 2021.

[3] W. Erwin Diewert, "The New Palgrave Dictionary of Economics," Palgrave Macmillan UK, 27 April 2017. [Online]. Available: https://link.springer.com/referenceworkentry/10.1057/978-1-349-95121-5_659-2. [Accessed 10 01 2022].

[4] S. Boyd and L. Vandenberghe, Convex Optimization, vol. 7, Cambridge University Press, 2009.

[5] S. Amaran, N. Sahinidis, B. Sharda and S. Bury, "Simulation Optimization: A Review of Algorithms and Applications," *Annals of Operations Research,* vol. 240, pp. 351-380, 2016.

[6] R. Nance and R. Sargent, "Perspectives on the Evolution of Simulation," *Operations Research,* vol. 50, no. 1, pp. 161-172, 2002.

[7] A. Juan, J. Faulin, S. Grasman, M. Rabe and G. Figueira, "A review of Simheuristics: Extending Metaheuristics to deal with Stochastic Combinatorial Optimization Problems," *Operations Research Perspectives,* vol. 2, pp. 62-72, 2015.

[8] K. Sörensen and F. Glover, "Metaheuristics," 2010.

[9] JaamSim, "2017 JaamSim Software Inc.," 2019. [Online]. Available: https://jaamsim.com/. [Accessed 17 March 2020].

[10] F. Glover, M. Laguna and J. Kelly, "SimWrapper by OptTek," 2021. [Online]. Available: https://www.opttek.com/products/optquest/simwrapper/. [Accessed 09 11 2021].

[11] P. K. Davis, "Generalizing Concepts and Methods of Verification, Validation and Accreditation (VV&A) for Military Simulations," *National Defense Research Institute,* pp. 5-6, 1992.

[12] A. M. Law, "How to Build Valid and Credible Simulation Models," 2019.

[13] J. P. Kleijnen, "Theory and Methodology of Verification and validation of simulation models," *European Journal of Operational Research,* vol. 82, pp. 145-162, 1995.

[14] F. Gortazar, A. Duarte, M. Laguna and R. Martí, "Black Box Scatter Search for General Classes of Binary Optimization Problems," *Computers and Operations Research,* vol. 37, no. 11, pp. 1977 - 1986, 2010.

[15] V. Campos, M. Laguna and R. Martí, "Context-Independent Scatter and Tabu Search for Permutation Problems," *INFORMS Journal on Computing,* vol. 17, no. 1, pp. 111 - 122, 2005.

[16] H. Hunter-Zinck, A. de Siqueira, V. Vásquez, R. Barnes and C. Martinez, "Ten Simple Rules on Writing Clean and Reliable Open-Source Scientific Software," *PLOS Computational Biology,* pp. 1 - 9, 2021.

[17] M. Jeong, J. H. Choi and B. H. Koh, "Performance evaluation of modified genetic and swarm-based optimization algorithms," *Structural Control and Health Monitoring,* p. 878–889, 2013.

[18] J. Shen and Y. Zhu, "Chance-Constrained Model for Uncertain Job Shop Scheduling Problem," *Soft Computing - A Fusion of Foundations, Methodologies & Applications.,* vol. 20, no. 6, pp. 2383-2391, June 2016.

[19] N. Gunantara and Q. Ai, "A review of multi-objective optimization: Methods and its applications.," *Cogent Engineering,* vol. 5, no. 1, pp. 1 - 16, 2018.