

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

GRADO EN INGENIERÍA ELECTROMECÁNICA

Especialidad Mecánica

Dispatching Material Handling Vehicles

Autor: Vela Tulupov, Yuri

Director: W. Herrmann, Jeffrey

Madrid

Junio 2019

Anexo H

AUTHORIZATION FOR DIGITALIZATION, STORAGE AND DISSEMINATION IN THE NETWORK OF END-OF-DEGREE PROJECTS, MASTER PROJECTS, DISSERTATIONS OR BACHILLERATO REPORTS

1. Declaration of authorship and accreditation thereof.

The author Mr. /Ms. YURI VELA TULUPOV

HEREBY DECLARES that he/she owns the intellectual property rights regarding the piece of work: DISPATCHING MATERIAL HANDLING VEHICLES
that this is an original piece of work, and that he/she holds the status of author, in the sense granted by the Intellectual Property Law.

2. Subject matter and purpose of this assignment.

With the aim of disseminating the aforementioned piece of work as widely as possible using the University's Institutional Repository the author hereby **GRANTS** Comillas Pontifical University, on a royalty-free and non-exclusive basis, for the maximum legal term and with universal scope, the digitization, archiving, reproduction, distribution and public communication rights, including the right to make it electronically available, as described in the Intellectual Property Law. Transformation rights are assigned solely for the purposes described in a) of the following section.

3. Transfer and access terms

Without prejudice to the ownership of the work, which remains with its author, the transfer of rights covered by this license enables:

a) Transform it in order to adapt it to any technology suitable for sharing it online, as well as including metadata to register the piece of work and include "watermarks" or any other security or protection system.

b) Reproduce it in any digital medium in order to be included on an electronic database, including the right to reproduce and store the work on servers for the purposes of guaranteeing its security, maintaining it and preserving its format.

c) Communicate it, by default, by means of an institutional open archive, which has open and cost-free online access.

d) Any other way of access (restricted, embargoed, closed) shall be explicitly requested and requires that good cause be demonstrated.

e) Assign these pieces of work a Creative Commons license by default.

f) Assign these pieces of work a HANDLE (*persistent* URL). by default.

4. Copyright.

The author, as the owner of a piece of work, has the right to:

a) Have his/her name clearly identified by the University as the author

b) Communicate and publish the work in the version assigned and in other subsequent versions using any medium.

c) Request that the work be withdrawn from the repository for just cause.

d) Receive reliable communication of any claims third parties may make in relation to the work and, in particular, any claims relating to its intellectual property rights.

5. Duties of the author.

The author agrees to:

a) Guarantee that the commitment undertaken by means of this official document does not infringe any third party rights, regardless of whether they relate to industrial or intellectual property or any other type.

b) Guarantee that the content of the work does not infringe any third party honor, privacy or image rights.
c) Take responsibility for all claims and liability, including compensation for any damages, which may be brought against the University by third parties who believe that their rights and interests have been infringed by the assignment.
d) Take responsibility in the event that the institutions are found guilty of a rights infringement regarding the work subject to assignment.

6. Institutional Repository purposes and functioning.

The work shall be made available to the users so that they may use it in a fair and respectful way with regards to the copyright, according to the allowances given in the relevant legislation, and for study or research purposes, or any other legal use. With this aim in mind, the University undertakes the following duties and reserves the following powers:

a) The University shall inform the archive users of the permitted uses; however, it shall not guarantee or take any responsibility for any other subsequent ways the work may be used by users, which are non-compliant with the legislation in force. Any subsequent use, beyond private copying, shall require the source to be cited and authorship to be recognized, as well as the guarantee not to use it to gain commercial profit or carry out any derivative works.
b) The University shall not review the content of the works, which shall at all times fall under the exclusive responsibility of the author and it shall not be obligated to take part in lawsuits on behalf of the author in the event of any infringement of intellectual property rights deriving from storing and archiving the works. The author hereby waives any claim against the University due to any way the users may use the works that is not in keeping with the legislation in force.
c) The University shall adopt the necessary measures to safeguard the work in the future.
d) The University reserves the right to withdraw the work, after notifying the author, in sufficiently justified cases, or in the event of third party claims.

Madrid, on 16..... of JUNE / 2019....., ...

HEREBY ACCEPTS

Signed.....Yuri.....Vela.....Tulyev.....

Reasons for requesting the restricted, closed or embargoed access to the work in the Institution's Repository

Anexo I

I, hereby, declare that I am the only author of the project report with title:

Dispatching Material Handling Vehicles

which has been submitted to ICAI School of Engineering of Comillas Pontifical University in the academic year 2018/19. This project is original, has not been submitted before for any other purpose and has not been copied from any other source either fully or partially. All information sources used have been rightly acknowledged.

Fdo.: Yuri Vela Tulupov Date: 16/6/2019

I authorize the submission of this project

PROJECT SUPERVISOR

Jeffrey W. Herrman

Fdo.: Jeffrey W. Herrmann Date: 17 June 2019

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

GRADO EN INGENIERÍA ELECTROMECÁNICA

Especialidad Mecánica

Dispatching Material Handling Vehicles

Autor: Yuri Vela Tulupov

Director: Jeffrey W. Herrmann

Madrid

Junio 2019

DISPATCHING MATERIAL HANDLING VEHICLES

Autor: Vela Tulupov, Yuri

Directores: Herrmann, Jeffrey

Entidad Colaboradora: ICAI - Universidad Pontificia Comillas

University of Maryland

Los vehiculos de reparto de material son ampliamente usados hoy en día. Estos, están dedicados al transporte de material entre dos puntos dentro de una fabrica, por lo que sus operaciones y requerimientos son algo distintos a los de los vehículos que estamos acostumbrados. Algún ejemplo de estos tipos de vehículos son las carretillas elevadoras. La optimización de el funcionamiento de estos vehículos es crucial para la mejora en cuanto a tiempos y dinero (gasolina, kilometraje…). A veces, la optimización de estos puede llevar a una mejora de tiempos de hasta un 30%. Viendo este dato desde una perspectiva anual, esto puede llevar a una mejora de productividad y de la eficiencia general de, por ejemplo, una planta industrial. Dentro de este tipo de vehiculos, se diferencia principalmente entre dos: los guiados manualmente (GVs) y los guiados mediante autómatas (AGVs). Normalmente, la inversión inicial en este último tipo de vehiculos es mayor que en los guíados manualmente, pero a lo largo del tiempo está inversión se ve amortizada ya que en los otros vehículos existe un coste de operario. Tipicamente, la decisión entre un vehículo u otro depende de las tareas a realizar por este, ya que los guiados manualmente son capaces de realizar tareas más complicadas. En este proyecto voy a implementar y evaluar el rendimiento de diferentes reglas de reparto adaptadas a este tipo de vehículos. A partir de estas evaluaciones se podrá determinar qué elementos afectan al rendimiento de cada una de estas reglas de reparto. Diferentes escenarios serán creados durante el proyecto. El primero de estos, es una fábrica localizada en España, y los siguientes dos escenarios son derivaciones a partir de este.

DISPATCHING MATERIAL HANDLING VEHICLES

Author: Vela Tulupov, Yuri

Directors: Herrmann, Jeffrey

Collaborating Entity: ICAI - Universidad Pontificia Comillas

University of Maryland

Material Handling Vehicles are widely used in the industry nowadays. These vehicles transport material, usually between two points inside a plant, so their operations and requirements are slightly different to the vehicles we are used to. Fork lifts are a good example of these type of vehicles. As these vehicles transport loads between different stations in a factory, the optimization of its work is crucial, not only in terms of time, but also in terms of costs (fuel, mileage…). Sometimes this optimization can lead to savings up to 30% of the waiting times, which escalated into an annual time save, can lead to the increase of productivity and overall efficiency of, for example, an industrial plant. These vehicles can be Automated (AGVs) or Guided (GVs). While AGVs are more expensive than traditional GVs, these last ones need a driver, which at the end, increases the costs. Usually, deciding between these two is done based on the tasks they are supposed to perform, as GVs can perform more difficult tasks. In this project I will be implementing and evaluating several dispatching rules in order to see what conditions make these dispatching rules work better or worse. I will be testing and tracking the performance of these rules in different scenarios, with some changes between them. The first scenario simulated in this is a real-world factory located in Spain, and the other two scenarios will be derivations of these first one.

Index

Table of Contents

Abstract ***11***

Introduction ***13***

Objectives ***15***

Background ***17***

Notation and assumptions ***19***

Description of how the factory works 19

Systematic Procedures and Algorithms ***33***

Procedure for layout creation 33

Procedure for implementing the functioning to the facility 35

Procedure for dispatching rules implementation 38

Experimental Results ***41***

First layout scenario 41

Second layout scenario 44

Third layout scenario 46

Results Comparison ***51***

First layout scenario 51

Second layout scenario 54

Third layout scenario 57

Summary ***61***

Conclusion and final thoughts ***67***

Appendices ***69***

Appendix A. Code for first layout scenario 69

Appendix B. Code for second layout scenario. 72

Appendix C. Code for third layout scenario. 78

Appendix D. Work List sample for Nearest Workstation First 87

Appendix E: Work List sample for Nearest Workstation First with Time Priority 88

Appendix F. Nearest Vehicle First code 89

Appendix G. Consensus Based Auction Algorithms applied to vehicle scheduling code 90

References ***..... 93***

Abstract

Dispatching rules are widely used in real industry in places such as warehouses, airports, distribution centers, etc. where the optimization of processes is crucial for the efficiency and strongly takes part in the cost of operation. In these facilities, Guided Vehicles (GV) are widely used to transport loads between different locations. These Guided Vehicles can be usually seen as forklift trucks or Automated Guided Vehicles (AGV's). Same dispatching rules can be implemented to AGV's as to vehicles guided by humans, the difference lies in the method of communication between vehicles. While AGV's communicate through algorithms programmed for each case scenario, GV's communicate through radio-frequency terminals attached to each vehicle, and a central station is in charge of the decision making. In this project the implementation of several dispatching rules to different factory layouts will be analyzed. Also, their performance will be tracked in order to see the benefits and drawbacks of each dispatching rules.

Introduction

Guided vehicles performance strongly affects the overall performance of a factory, which makes them an essential part to take into account when designing facilities. Their performance is directly related with dispatching rules, which select what task a vehicle should perform, which load each vehicle should claim… These dispatching rules are mainly based in three related variables: distance, velocity and time. In this paper, no distinction between AGVS and GVs will be made.

In this project, three different layout scenarios have been created. These three scenarios are mainly derivations of the original one, which is a real-world factory located in Spain. Data used to simulate the plant was facilitated by the company, and it will be analyzed in this paper. Also, some assumptions, which also will be reviewed later, were made in order to create the simulation.

The main purpose of modifying the original layout is to track the performance of each dispatching rule in larger scenarios, where more tasks need to be completed, more loads are generated, and working times, velocities and distances will change too. This will lead to more complex decision making and conflicts between vehicles to decide which one is performing each job.

In order to achieve maximum efficiency of the plant, the main objective of the dispatching rules applied is to reduce load waiting times to the minimum possible and also, have vehicles and resources (workers) working at its maximum capability.

For the simulation AutoMod will be used. AutoMod is a simulation software, widely used in industry, which enables the creation of all type of facilities with the main purpose of optimizing them. This program also has an advanced statistics generator, which facilitates the comparison between results.

Objectives

In this section a description of the objectives of this project will be made. Some of these objectives are meant to be achieved through the process of the elaboration of this project and others are expected to be achieved at the end of the project.

One of the objectives is the apprenticeship of the realization of a technical paper. This objective encompasses the realization of research, simulation and also drawing conclusions from data obtained. Also, the presentation of a project and being well prepared to answer different questions about it are objectives that are meant to be achieved. This objective is meant to be completely achieved by the end of the realization of the project.

As a more specific goal the application of optimization techniques arises. Through the development of the project different techniques will be applied. Some of these techniques are the dispatching rules or the path approach taken in each case. These techniques will be evaluated and compared.

Another objective is the development of logistic algorithms through coding. Some algorithms will be coded in the language of programming of the used software, AutoMod.

The familiarization with logistic software's such as AutoMod also is an important goal for this project. These types of software's are widely used in the industry nowadays and knowing how to work with them is an important attribute.

Finally, the application of the mentioned before tools in a real factory will give me the insight of how vehicles are dispatched in the industry. Also, working with layouts gives an idea of how factories are set up.

Background

This research paper is mainly aimed to logistics managers of facilities where material handling between locations inside the same facility is approached by the use of Guided Vehicles. Its usefulness could also relate to engineers in charge of optimization of the process of a factory.

Literature classifies dispatching rules into two types, depending on the operation decisions: Vehicle-initiated dispatching and load-initiation dispatching. In vehicle-initiated dispatching when a vehicle becomes idle it executes the assigned procedures in order to find its next job. On the other hand, in load-initiation dispatching the load is the one in charge of deciding which vehicle has to pick it up. Putting it clear, when a load arises in the system, procedures are executed to assign it to a vehicle.

Actually, literature analyzing dispatching rules and its performance is not easy to find. Most of the articles reviewed before and through the process of work are dated back in the 1990's. The main articles in which this project takes its base are:

- In *"On-line Dispatching Rules", T. LE-ANH and M. B M. DE KOSTER* analyze Single-Attribute Dispatching rules as well as Multi-Attribute dispatching rules. This paper simulates the dispatching rules in AutoMod.
- In "Development of Dispatching Rules for Automated Guided Vehicle Systems", Hark Hwang and Sang Hwi Kim incorporate the idea of bidding functions applied to the selection of jobs for vehicles.
- In "Consensus-Based Auction Approaches for Decentralized Task Assignment" Luc Brunet, Han-Lim Choi, and Jonathan P. How present the incorporation of Consensus-Based Auction Approaches to job selection procedures.

In this project some of the dispatching rules present in these articles will be applied. Also, the incorporation of CBAA to Guided Vehicles and its later simulation in AutoMod will be present.

Notation and assumptions

In order to be able to simulate the facility and the dispatching rules selected, some assumptions were made. First, a brief description of the working procedure of the facility simulated is going to be provided.

Description of how the factory works

This factory dedicates its activity to the improvement of the properties of metallic thread coils. Broadly, the process of the factory is the following: costumer provides the empty metallic thread coils and the company, by a pressing process, modifies coils in order to ensure its good behavior for later operations.

As it was previously stated, the data was obtained from the company in charge of the facility. They provided a scheme of the plant as well as the times each of the steps take. Attached below is the facility scheme.

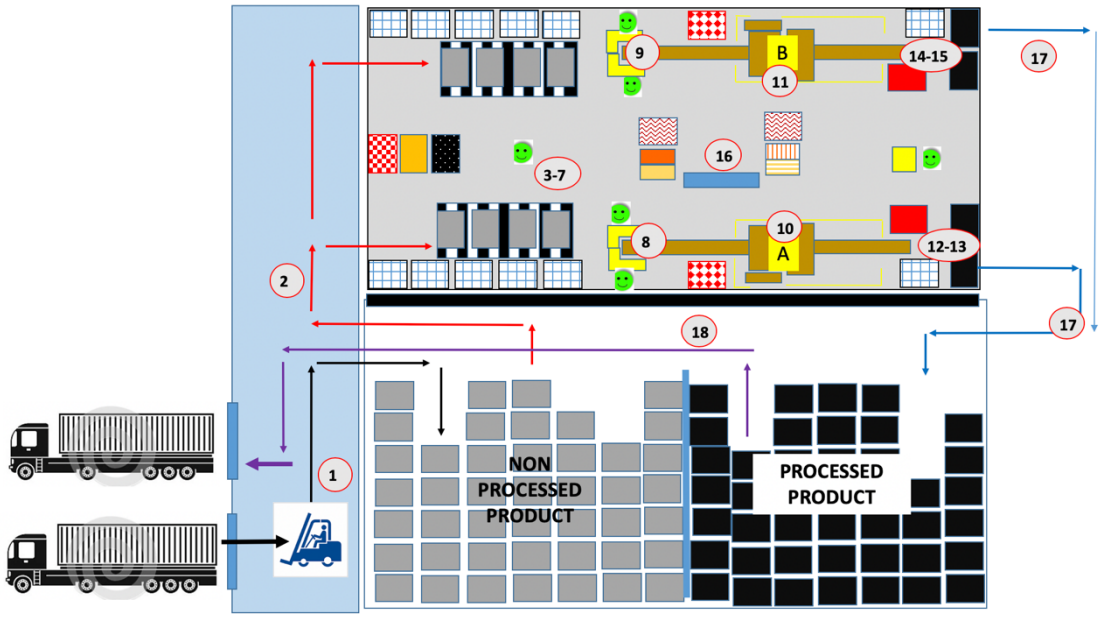


Figure 1: "Facility scheme" provided by [HERA19]

The operation flow is the following: A truck containing pallets, in which coils are stored, enters the plant (1). Each of the pallets has to be transported to the non-

processed-product storage area. In this area pallets wait until they are moved to the process area (2), in which depalletization, pressing and again palletization occurs (3-15). Pallets, already containing pressed coils, are transported to the processed-product storage (17), and finally are moved to the leaving truck (18). Times for each operation also were provided by the company and are stated in the chart below.

Nº	Operation	Time (seconds)
1	Truck unloading	75 s / palet
2	Move pallets to storage area	Undef
3	Depalletization	20s / pallet
4		
5	QA 20 coils	30 s / 20 coils
6	Move 8 coils to work station A & B	16 s / 8 coils
7		
8	Pressing process	45s / coil
9		
10		
11		
12	Palletization	8 s / 2 coils
13		30 s / pallet
14		8 s / 2 coils
15		30 s / pallet
16		60 s / pallet
17	Move pallets to storage area	undef
18	Truck loading	75 s / pallet

Figure 2: "Time per task table"

One truck arrives at the factory per day and each of these trucks has 32 pallets with 80 coils per pallet.

Vehicles used in the factory are conventional fork lifts driven by qualified workers. The company uses 2 vehicles to complete the tasks. Data about vehicles parameters was not provided. The path used by vehicles to move from one point to another was not specified by the company. The company specified that a parking station is present in the factory.

Assumptions for the model

After presenting how the factory works and the data provided by the company, in in this section this project will explain how this data was used in order to create the model in which simulations will be run.

As path was not specified, the approach for its creation was having minimum distances between locations. For that, a loop approach with shortcut paths was selected. Also, for each task there is a station, which is not located in the main path, in order to avoid having retentions. Also, as the company indicated, a parking station was added in each of the layouts.

The time for each operator working in a task was taken from the chart. Some tasks were combined, which means that an operator was set to perform more than one task. These times will be stated later.

There will be three main parameters that are the ones which will be changed in each simulation. These parameters are:

- Variation of load creation ratio.

As mentioned before, each of the trucks arriving has 32 pallets. This value will change in order to create variations and better evaluate the dispatching rules.

- Variation in the number of coils per pallet.

This number will be adapted in order to try not to saturate operators, as long as possible.

In the following paragraphs the assumptions taken in this project will be more deeply analyzed. First, assumptions valid for the three layout scenarios will be analyzed and after that the particular ones for each case.

- General assumptions for all the cases.

As mentioned earlier, path was created in order to have minimum distances between locations. Also, parking stations have been created, where vehicles can park only when idle.

For simulating the pressing process, a conveyor was used. The way it is set up is that it takes the same time for a load to go through the conveyor as it would take the same load to go through the pressing process. This conveyor has unlimited capacity. Storage areas are simulated as queues. These queues are set to have unlimited capacity.

A load generation rate has been set up simulating the arrival of a truck. Loads arriving at the Truck-In station (where trucks arrive) are pallets, which contain the coils that go through the process and then are stored again in pallets. In order to visually distinguish them, pallets have different dimensions that coils in the simulation.

Stock storage areas are set up as queues. This means that, after a determinate time after a load enters in the storage area it is ready to leave it.

- First layout scenario

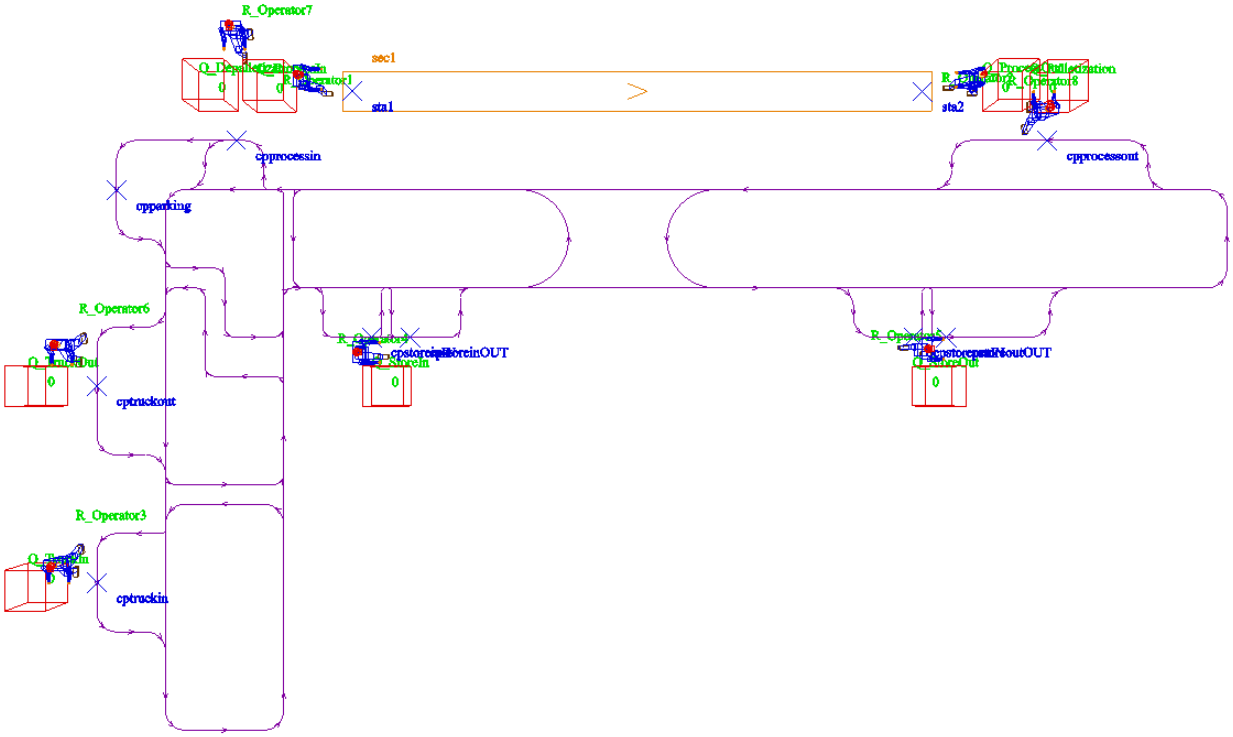


Figure 3: "First layout scenario"

First layout scenario simulates the real-world factory. The number of workers is set up to eight. The time it takes to perform each job is presented in the presented before. The parking location in this case has been located in the upper left corner.

- Second layout scenario

The second layout scenario incorporates an additional "process line" simulated as a conveyor, as before. It also incorporates four more operators and additional paths with some shortcuts.

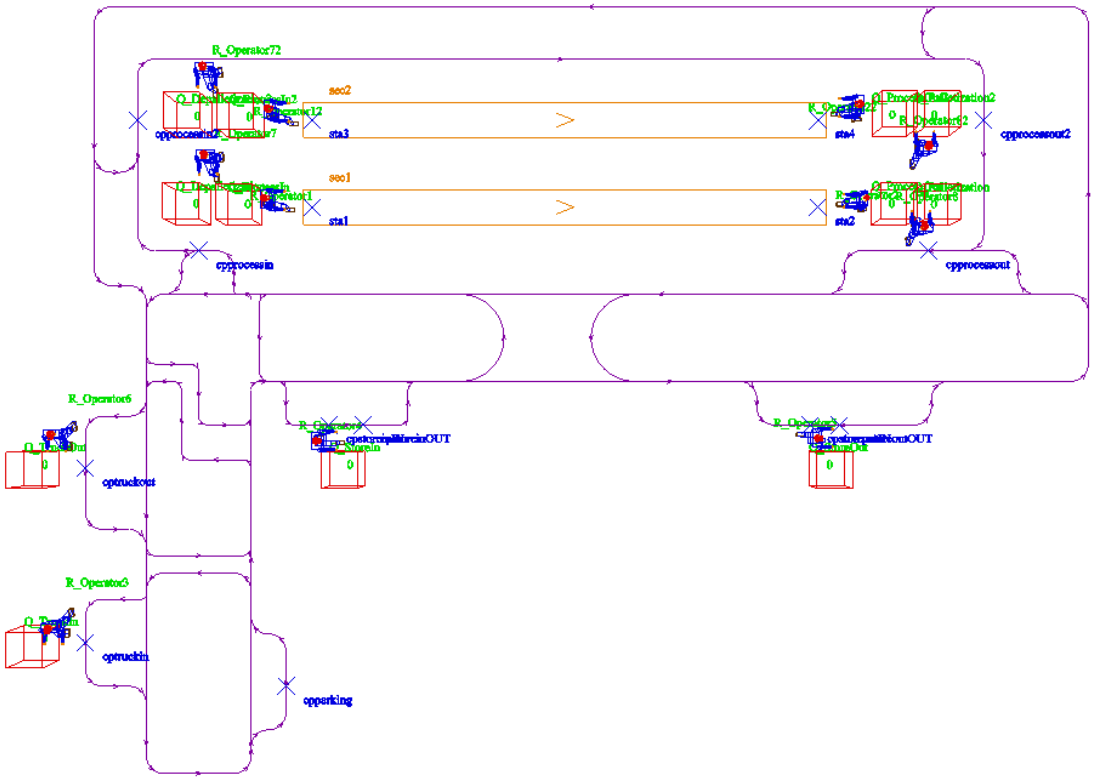


Figure 4: "Second layout scenario"

Operators times for this scenario are the same as in previous simulations. The additional operators added in this scenario have the same operation times as the ones in the "original process line".

- Third layout scenario

For this scenario the disposal of the stations was changed. Also, an additional check station was created. These station checks whether the pallet is in good condition. The rejection rate was set up to 10%.

Another difference when comparing this scenario with previous ones is the way the depalletization and palletization tasks are done. While in previous scenarios each "process line" had its own depalletization and palletization stations, in this scenario all three lines share the depalletization and palletization stations.

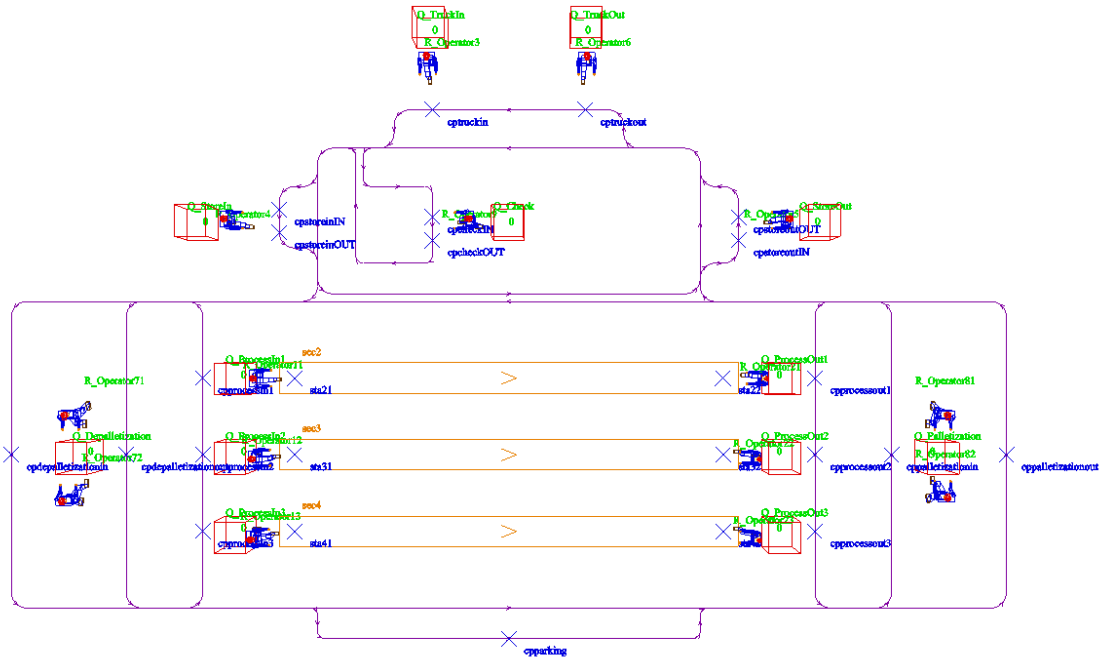


Figure 5: "Third layout scenario"

As in previous scenario, operators' times per tasks are the same as in the previous scenarios. In this case we have fifteen operators.

Dispatching rules

In this section the dispatching rules implemented in the model will be described.

Nearest Workstation First [TANC94]

In this "vehicle-initiated" dispatching rule, when a vehicle becomes idle it searches for the closest available load to transport. The distance is measured as the path distance from the position of the vehicle to the available load position.

The major disadvantage of this rule takes place when a factory has a remote station, as in most cases the rule indicates to go for another load before going to the remote one.

Nearest Vehicle First [TANC94]

In this “load-initiated” dispatching rule, when a new load arises in the system, it searches for the closest vehicles available to transport it. Again, the distance is measured as the path distance from the position of the vehicle to the available load position.
The major withdrawal of this rule is the same as of the previous one, the remote stations in a layout, if present.

Nearest Workstation First with Time Priority

The main objective of this rule is to decrease the impact of the remote locations in the performance of the “Nearest Workstation First” dispatching rule.
As mentioned in the simple Nearest Workstation First dispatching rule, the major disadvantage of NWF were the remote stations present in the factory layout. While the main idea of the dispatching rule is the same, it incorporates a threshold for the load waiting time, so that, when a load waiting time exceeds the boundary value it becomes the priority. Because of its importance, selecting the threshold value should be a cautious process. If threshold value is too large, the performance will be the same as for the “Nearest Vehicle First”. On the other hand, if the threshold value is too low, it will worsen the performance of the “Nearest Vehicle First”, as the remote load will have preference too fast. The threshold value in this project is selected as four times the average waiting time per load per whole process.

Consensus Based Auction Algorithms applied to Vehicle Scheduling [HAFI19]

This is another approach to vehicle scheduling. In this project, “CBAA” will be used for selecting which vehicle has to perform each job. CBAA has mainly three different parts. First, the Bid Generation Problem, where bids are placed. Second, the Winner Selection Problem, where winner is selected, and third, the Auctioning Rules.
First part is the bid generation problem. When a load appears in the system, each of the available vehicles “place a bid” depending on basically three related-between-them variables. These variables are velocity, time and distance. The objective function is:

$$\text{Minimize } \text{at}_{ij}^{x} \cdot \beta_{ij}^{x} + \sum_{i=1}^{I} \sum_{j=1}^{J} uc \cdot d_{ij}^{x} \cdot \beta_{ij}^{x}$$

- $a\,t^{x}{}_{ij}$ Expected arrival time of vehicle x for operation ij
- $\beta^{x\,ij}$ One if vehicle x is assigned task ij, zero if not
- UC Unit cost (which is an inverse of the velocity time/space)
- d_{ij}^{x} Shortest node to node distance to transport load ij

Constraints and expected arrival calculations are the following:

$$at_{ij}^{x} = tr_{ij}^{x} + ct_{x} \tag{2}$$

$$ct_{x} = \sum_{i=1}^{I}\sum_{j=1}^{J}\left[\left(tta_{ij}^{x} + (tr_{ij}^{x} + td_{ij}) + (tv_{ij} + tw_{ij})\right)\beta_{ij}^{x}\right] \tag{3}$$

$$\beta_{ij}^{x} = 1 \tag{4}$$

$$cc_{m}^{x} < cp_{x} \tag{5}$$

$$cc_{m}^{x} \geq 0 \tag{6}$$

$$cc_{m}^{x} + cl_{m}^{x} \leq \sum_{m\in S}^{S} \delta_{m}^{x} \cdot cp_{x} \tag{7}$$

$$ce_{m}^{x} = 1 \Rightarrow cc_{m}^{x} + cl_{m}^{x} = cc_{n}^{x} \tag{8}$$

$$\sum_{mn\in L}^{L} \delta_{mn}^{x} - \sum_{no\in L}^{L} \delta_{no}^{x} = 0 \tag{9}$$

$$\sum_{p_{ij}\, m\in L}^{L} \delta_{p_{ij}m}^{x} - \sum_{ne_{ij}\in L}^{L} \delta_{ne_{ij}}^{x} = 0 \tag{10}$$

$$\sum_{p_{ij}\, m\in L}^{L} \delta_{p_{ij}m}^{x} = 1 \tag{11}$$

$$\sum_{ne_{ij}\in L}^{L} \delta_{ne_{ij}}^{x} = 1 \tag{12}$$

$$\sum_{mn\in L}^{L} \delta_{mn}^{x} = 1 \tag{13}$$

$$\sum_{mn\in L} \delta_{mn}^{x} + \sum_{mn\in L} \delta_{nm}^{x} \leq 1 \tag{14}$$

$$\sum_{b\in B_x}^{B_x} \xi_{b}^{x} \geq 1 \tag{15}$$

- (2) calculates the expected arrival time.
- (3) calculates the time the vehicle x has left to finish performing the task it is actually performing.
- (4) ensures only assigned tasks are taking into account when performing these calculations.
- (5) makes sure the number of loads in board are less than the capacity.
- (6) loading capacity only takes positive values or 0.
- (7) ensures that loading and unloading take into account the capacity limitation.
- (8) binary decision variable considering vehicle capacity.

- (9-14) are constraints related to AGV routing.
- (9) ensures that when a vehicle enters a station through its incoming lane, it leaves this station through its outgoing lane.
- (10) makes sure that each of the loads a vehicle is carrying is carried to its final destination.
- (11) defines the starting node that an AGV needs to follow.
- (12) defines the destination node that an AGV needs to follow.
- (13) ensures that one AGV can only be located at one location in the path.
- (14) ensures that only one AGV can be in each path point at the same time, avoids collisions.
- ξ_b^x represents the status of the bid, 1 if it is optimal, 0 if not. (15) makes sure we have at least one optimal bid.

The next step after the bid generation problem is the winner selection problem. Vehicles get ranked depending on the bid each one places. The objective function and its constraints are the following:

$$\text{Minimize} \sum_{i \in I} \sum_{j \in J} ad_{ij}^x \cdot \xi_b^x(ij)$$

$$ad_{ij}^x = |te_{ij} - at_{ij}^x| \quad (17)$$

$$te_{ij} \leq at_{ij}^x \leq tl_{ij} \quad (18)$$

$$tl_{ij} = rt_j + dt_j - mt_j \quad (19)$$

$$te_{ij} = tp_{ij}\alpha_{ij}^m + tpa_{ij}\alpha_{ij}^m \quad (20)$$

$$tpa_{ij}^m \geq (tpa_{ij-1}^m + tp_{ij-1}^m + tv_{ij-1})\alpha_{ij-1}^m + tw_{ij} \cdot \alpha_{ij}^m \quad (21)$$

- The objective function minimizes the difference between the expected vehicle arrival time and earliest pick up time.
- (17) calculates the difference between the earliest pick up time and the vehicle arrival time.

- (18) ensures expected arrival time neither starts before earliest task pickup nor exceeds the latest start time.
- (19) calculates the latest start time for operation ij.
- (20) calculates the definition of earliest start time.
- (21) ensures that machines operations comply with the precedence sequence for operations belonging to the same job.

Finally, some auctioning rules have to be taken into account. These rules are applied to verify whether the selection is feasible and also, for example, it can check the "preference" of each bidder of getting that load or not. Other examples of this auctioning rules are for example that vehicles can bid for more than one load, or that all vehicles can carry all the load types.

Systematic Procedures and Algorithms

Procedure for layout creation

The procedure followed for creating each of the layouts present in this project will be described in this section.

1. Brainstorm of possible layout derivations out of the first one in order to achieve a new "simulation scenario". This step was followed in all scenarios except for the first one, which was already given by the factory.
2. Selection of the layout to implement. This selection was made based in three main objectives: First, the introduction of new job selection conflicts into the system in order to test dispatching rules at its maximum. Second, the feasibility of the layout scenario, as the software used was AutoMod's student version and the limit number of entities was 200. And third, layout created had to make sense. By make sense it is mean that each of the stations were located in order, as they are located in real-world.
3. Creation of the layout in AutoMod. This process encompassed:
    - Design of the conveyor system emulating the main process of the factory, which in this project was the pressing of coils.
        - Creation of the conveyor. The time for the process was adjusted with conveyor velocities, distance and capacity of the conveyor.
        - Creation of conveyor stations. Loads on the conveyor travel from one of these stations to another.
    - Creation of queues. These queues simulate the different stations the vehicles go through. All three scenarios have the same stations: Truck in, truck out, storage in (for non-processed product), storage out (for already processed product), process in (the number of these stations variate from one model to another), process out (the number of process out stations also variate between models) , depalletization (number of stations also variate) and palletization (also variates the number of stations). One

additional queue was created in simulation model number three: Check station. These queues are present in the layout scheme as boxes.

- Addition of workers to the model. AutoMod calls them “Resources”. Each of the workers was created with same specifications and same graphic. Also, each of the workers was placed near to the station where he is doing his job.
- Creation of loads. The creation of the three loads previously described was the same for the three different scenarios. Dimension and creation rate of loads was also specified. This creation rate was only specified for the first load, which simulates the arrival of the truck.
- Creation of the AGV system. The procedure followed was:
    - Creation of the path, which encompassed the following: Creation of the “main/first path” (loops). Creation of “second paths”. Second paths are the ones that are not present in the main loop and are not the station specific ones. These paths are mainly shortcuts. Creation of “third paths”. Third paths are the exits to the stations, called control points, which will be explained later. Fillet of these paths, in order to make the connection between paths curved. Otherwise, it takes more time for vehicles to make the turns.
    - Creation of control points. Present in schemes as “X”, they are the designated points in the path where vehicles stop to either pick up or deliver a load. The way these control points are set up is: Stations where only one task has to be performed, either deliver or pick up, have only one control point. Also, stations where both deliver and pick up tasks happen have two control points. Finally, the parking station is also created as a control point.
    - Creation of vehicles. Two types of vehicles were created for this project: *“AGVO”* and *“AGVI”*. Number of vehicles present, and its attributes, will be specified for each simulation.

Having the layout created, the very next step is implementing the "functioning of the facility". This is for example: assigning times for operations or assigning each vehicle where to transport each load. Next part of the project will explain it.

Procedure for implementing the functioning to the facility

In this section the project will go through the creation of the "non-visible" elements that make the simulation run.

The first step is the "**Creation of the processes".** Each of the actions taking place in the simulation has its particular process. First model incorporates 10 processes. Below is the description of these processes from the first, when truck arrives to the system, to the last, when it leaves:

Truck-In. Simulates the entrance of a truck to the system and transfers the pallets to the "Truck in" queue, where they wait to be picked up. Operator 3 is in charge of this task. Sends to "Store-In".

Store-In. This process simulates the transportation of pallets to the storage area. When a vehicle picks up the pallet it travels with it to the "Storage in" queue. Operator 4 moves the load to the queue. Sends to "Store-In-Out".

Store-In-Out. This process simulates the exit of the load from the non-processed product storage area. Operator 4 moves the load to the AGV system. Sends to "Depalletization".

Depalletization. This process simulates the transportation of pallets from non-processed product storage area to the "Depalletization" queue. When a vehicle with a pallet arrives, Operator 7 moves it into the depalletization queue. Operator 1 opens the pallet and extracts the coils contained in it. After that he puts these coils into "process in" queue (conveyor). Number of coils contained per pallet is set up here. Sends to "Enter assembly".

Enter assembly. Simulates the entrance to the pressing process. Operator 1 takes the load and transfers it from the "process in" queue to the initial conveyor station. Sends to "Complete assembly".

Complete assembly. Simulates the exit of the pressing process. Operator 2 takes the load and moves it from the final conveyor station to the "process out" queue. Sends to "Palletization".

Palletization. Simulates the process of putting the processed coils to a pallet. When the required number of coils to fill a pallet are ready in "process out" queue operator 8 fills a pallet and moves it into "palletization" queue. Sends to "Store-Out".

Store-Out. Simulates the process of transporting a pallet from the exit of the process to the processed product storage area. A vehicle picks up a pallet and transports it to the "store out" station. There, operator 5 moves the load to the "store out" queue. Sends to "Store-Out-Out".

Store-Out-Out. This process simulates the exit of a pallet from the processed product storage area. Operator 5 moves the load to the AGV system, where it is picked up by an AGV. Sends to "Truck-Out".

Truck-Out. Simulates the movement of pallets from the vehicles to the queue and later to the truck. This task is performed by operator 6.

The time it takes to perform each of these tasks for each of the operators is set up in each of the processes. Also, the statistical distributions used to simulate these times are set up inside these processes.

Simulations two and three have almost the same processes. The only difference is that some of them are doubled, because of for example, the addition of a process line. Also, in third scenario a check station is added, and incorporates a new process.

Check. It simulates a checking station. When loads pass through this station are checked in order to see whether they are in a good condition or not.

The second step is the **"Addition of blocks"**. Blocks ensure vehicles do not crash between them when operating. A total of 15 blocks were set up for first simulation.

The third step is the **"Creation of variables".** These variables are used for depalletization processes, counters, time in process of each load… The variables created were the following:

- A_Time. This time type variable measures the time a load is in the process.
- V_Palletization. This integer type variable is used to fill pallets out of processed coils. When a coil finishes the process this variable increases by 1, and when reaching the number of coils needed per pallet, a pallet is created. After that, the variable is re-initialized to 0, and so on.
- Variables created for dispatching rules implementation, which will be explained in the *"Procedure for dispatching rules implementation"* section.
- Variables created for tracking the time. Some variables were created in order to track waiting times and times per process.

After that, the next step is the **"Establishment of time per simulation"**. In this project all simulations are run up for 8 hours, which is a working shift in the factory simulated.

Finally, it is time to **"Creation of source file".** Here processes and dispatching rules are coded. The code for the processes is in the appendix.

Procedure for dispatching rules implementation

The procedure followed for implementing the dispatching rules which were explained earlier will be described in this section. All dispatching rules were implemented the same way in all three scenarios. Previous to the dispatching rule implementation explanation I will describe some AutoMod concepts needed to understand this section.

Two ways were used to schedule vehicles in AutoMod. While for some dispatching lists were used, for others coding was the method used. The way AutoMod operates is: When a vehicle becomes idle it first checks the Work List present in that station. It checks from the top of the list to the bottom until it finds an available task. The closest to the top a task is, the more importance it has. Work Lists are only present in stations where AGV's deliver loads, as these stations are the only ones where vehicles become idle. that, Coding permits canceling or reassigning scheduled jobs. In this project the dispatching rules implemented by coding are load-initiated, so, as it would be shown in the code the load searches and selects a vehicle.

- Nearest Workstation First. In order to implement this dispatching rule, AutoMod's Work Lists were used. First, path distances between stations were calculated for each scenario for later creating the Work List for each station. Control points were ordered from closest to the farthest in each of the station's Work Lists. The way a vehicle goes through these Work Lists is from the top to the bottom, so control points were ordered that way.
- Nearest Workstation First with Time Priority. The implementation of this dispatching rule is also done using AutoMod's Work Lists. The way it was done was, for each station, the same list as before was added but with the time priorities. So, when a vehicle goes through this Work List, if first looks for the stations with a load that has been waiting for more than λ (threshold value) and if no load passes these filters it runs the list specified

in NWF. Value λ has been set up as four times the average waiting time obtained when simulating NWF.

- Nearest Vehicle First. This rule was implemented by coding. A loop was created in order to add this dispatching rule to the program. This loop calculates the distances between vehicles location and the location at which a load has to be picked up. After that it selects the nearest vehicle available and sends messages to the display window indicating which vehicle was selected and the calculated distances for each vehicle. The code is attached in the appendix.
- Consensus Based Auction Algorithms applied to Vehicle Scheduling. This rule was also implemented by coding. The way this dispatching rule was implemented was by the creation of a loop that for every vehicle available in the facility at the time of the simulation, calculates a bid. Later, it compares each vehicles' bid and assigns one vehicle for the job.

  The lowest the bid, the better. The program also sends messages to the display window in which it indicates the bids and which vehicle was selected in each case. The full code is attached in the appendix of this project.

Experimental Results

After explaining the creation of the layout scenarios and the dispatching rules implementation, in this section results got for each case will be provided. As explained before, all dispatching rules were simulated in the three scenarios.

The main objective of the dispatching rules implemented is to achieve the minimum average load waiting time. Dispatching rules will be evaluated according to that goal. For this purpose, a variable called "Average Waiting time per Load" has been created. It sums the **waiting** time from the entrance to the exit of all loads, and then divides it by the total number of loads. The time it takes for a load to go through the whole process, from arriving at the "Truck-In" station, to leaving the facility at "Truck-Out", will be also considered. Other parameters such as vehicle utilization percentage or distance traveled by each vehicle will be considered.

Along this section, simulations will be referred to with numbers from one to nine, as different simulations numbers have the same parameters, but different dispatching rule as it is shown in the next paragraphs.

First layout scenario

In order to analyze the dispatching rules performance in this a total of eight simulations for each dispatching rule in each scenario will be done. Parameters for each simulation are attached in the following chart:

	Number of AGVs	Load generation rate	Loads / pallet	AGVs velocities
Simulation 1,9,17,25	2	13.33min + 60 sec	5	3 ft/sec
Simulation 2,10,18,26	2	13.33min + 60 sec	10	3 ft/sec
Simulation 3,11,19,27	2	13.33min + 60 sec	15	3 ft/sec
Simulation 4,12,20,28	2	6.67min + 30 sec	5	3 ft/sec
Simulation 5,13,21,29	2	6.67min + 30 sec	10	3 ft/sec
Simulation 6,14,22,30	2	6.67min + 30 sec	15	3 ft/sec
Simulation 7,15,23,31	2	5min + 22 sec	5	3 ft/sec
Simulation 8,16,24,32	2	5min + 22 sec	10	3 ft/sec

Figure 6: Data chart for first layout scenario

The data obtained for each dispatching rule simulation is attached below. As it can be noticed in the chart below some "Average Time in Process" are much higher than others. That is because of the saturation of some operators in the process. Even so, this saturation only affects in the "Average Time in Process" and does not affect to the vehicles data.

NWF	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 1	174,85	1231,55	24,02	18626,45
SIM 2	183,51	1409,91	24,76	19032,5
SIM 3	101,33	1612,91	24,81	18907,85
SIM 4	48,13	1231,51	49,25	38287,6
SIM 5	55,35	1710,22	48,09	37298,75
SIM 6	55,22	6174,41	37,93	28960,75
SIM 7	27,16	1247,47	58,57	45913,35
SIM 8	23,50	4959	78,29	22203,5

NWF w/ time priority	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 9	78,02	1526,76	17,86	18547,05
SIM 10	80,24	1945,25	17,53	18211,6
SIM 11	115,86	1556,98	18,55	18994,1
SIM 12	93,31	1162,95	36,70	38072,9
SIM 13	46,01	1678,6	36,34	37675,55
SIM 14	54,12	6104,28	29,12	29140,8
SIM 15	54,58	1169,49	49,11	51517,65
SIM 16	23,68	4922,06	40,96	41830,6

NVF	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 17	121,07	1243,08	17,85	18782,9
SIM 18	108,68	1815,94	18,17	18752,45
SIM 19	105,64	1840,72	18,55	18853,6
SIM 20	68,93	1236,97	35,865	37549,8
SIM 21	80,1	1721,06	35,83	36848,35
SIM 22	81,96	6171,69	29,38	29139,95
SIM 23	67,11	1202,14	47,23	49729,45
SIM 24	51,29	4817	40,105	40658,5

CBAA	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 25	112,28	2071,72	16,06	16141,05
SIM 26	149,83	2476,81	16,90	17349,75
SIM 27	151,65	2811,78	16,61	16951,55
SIM 28	115,19	1725,17	33,61	34678,25
SIM 29	96,05	2439,89	31,71	31866,05
SIM 30	123,68	6684,20	27,17	27379,53
SIM 31	52,94	1798,23	43,26	43149,25
SIM 32	48,28	5383,89	37,65	37370,3

Figure 7: Simulation results for First Layout Scenario

Second layout scenario

As stated previously, the main difference between the first and second layout scenarios is the addition of a second "process line" in the second scenario. This "process line" is simulated as a conveyor. Attached below are the values used for the several simulations performed in this scenario. In the "Loads/pallet" column the first value indicates the number of loads per pallet in the first conveyor and the second figure indicates the number of loads per pallet the second conveyor operates with. Nine simulations were performed for each dispatching rule.

	Number of AGVs	Load generation rate	Loads / pallet	AGVs velocities
Simulation 1,10,19,28	2	13.33min + 60 sec	5/10	3 ft/sec
Simulation 2,11,20,29	2	13.33min + 60 sec	10/15	3 ft/sec
Simulation 3,12,21,30	2	13.33min + 60 sec	5/15	3 ft/sec
Simulation 4,13,22,31	2	6.67min + 30 sec	5/10	3 ft/sec
Simulation 5,14,23,32	2	6.67min + 30 sec	10/15	3 ft/sec
Simulation 6,15,24,33	2	6.67min + 30 sec	5/15	3 ft/sec
Simulation 7,16,25,34	2	5min + 22 sec	5/10	3 ft/sec
Simulation 8,17,26,35	2	5min + 22 sec	10/15	3 ft/sec
Simulation 9,18,27,36	2	5min + 22 sec	5/15	3 ft/sec

Figure 8: Data chart for second layout scenario.

The results got after simulating these values were the following:

NWF	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 1	94,22	411,66	18,10	25469,1
SIM 2	78,36	1907,14	17,85	24743,6
SIM 3	84,33	1727,66	17,63	24526,4
SIM 4	58,45	4243,92	35,47	49606,8
SIM 5	45,41	3284,88	35,36	49344,55
SIM 6	75,34	2463,14	33,48	45456,8
SIM 7	59,57	2648,48	45,65	62612,1
SIM 8	53,98	4415,44	43,47	59185,55
SIM9	60,03	2928,86	42,55	57443,85

NWF w/ time priority	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 10	88,94	408,58	17,19	23670,95
SIM 11	78,36	1907,14	17,85	24743,6
SIM 12	84,33	1727,66	17,63	24526,4
SIM 13	88,03	3432,81	36,64	51031,2
SIM 14	50,79	2172,69	24,07	33527,4
SIM 15	70,16	2085,62	33,99	46435,3
SIM 16	58,76	3206,75	45,55	61829,85
SIM 17	48,43	2926,10	44,29	60462,3
SIM 18	53,78	2373,17	43,90	60338,1

NVF	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 19	97,79	4237,50	15,53	21387,6
SIM 20	66,29	554,08	18,11	24529,4
SIM 21	99,78	2982,10	16,29	22494,85
SIM 22	66,97	2411,80	33,37	45610,7
SIM 23	76,09	3408,16	33,64	46088,35
SIM 24	68,89	2129,86	32,14	43829,7
SIM 25	56,87	1669,14	40,31	54431,65
SIM 26	64,53	3452,35	42,08	57040,65
SIM 27	57,90	1689,48	41,223	56174,7

CBAA	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 28	119,30	4766,35	16,95	23049
SIM 29	65,72	535,96	18,11	24529,4
SIM 30	107,68	3414,16	17,62	24132,05
SIM 31	57,40	264,87	35,47	48922
SIM 32	67,43	3274,84	35,42	48561,05
SIM 33	68,81	3227,48	34,01	46111,25
SIM 34	62,58	3232,90	44,55	59588,6
SIM 35	50,81	3663,40	42,78	57508,95
SIM 36	60,17	3386,05	43,80	58735,45

Figure 9: Simulation results for Second Layout Scenario

Third layout scenario

Third layout scenario is completely different. The "loop" approach used for designing the path causes the dispatching rules to work in a slightly different way compared with the previous scenarios. As previously described, the only difference in terms of "stations" is the addition of a new "Check" station. Other stations were replaced and the number of "process lines" present in this scenario is three, which also elevates the number of operators. Attached below are the used values for each simulation. As in the previous scenario, loads/pallet indicate the number of loads in the first, second and third "process lines" respectively. Also, nine simulations were performed.

	Number of AGVs	Load generation rate	Loads / pallet	AGVs velocities
Simulation 1,10,19,28	2	13.33min + 60 sec	5/6/7	3 ft/sec
Simulation 2,11,20,29	2	13.33min + 60 sec	8/9/10	3 ft/sec
Simulation 3,12,21,30	2	13.33min + 60 sec	3/4/5	3 ft/sec
Simulation 4,13,22,31	2	6.67min + 30 sec	5/6/7	3 ft/sec
Simulation 5,14,23,32	2	6.67min + 30 sec	8/9/10	3 ft/sec
Simulation 6,15,24,33	2	6.67min + 30 sec	3/4/5	3 ft/sec
Simulation 7,16,25,34	2	5min + 22 sec	5/6/7	3 ft/sec
Simulation 8,17,26,35	2	5min + 22 sec	8/9/10	3 ft/sec
Simulation 9,18,27,36	2	5min + 22 sec	3/4/5	3 ft/sec

Figure 10: Data chart for third layout scenario

Below are attached the results got for each dispatching rule after applying the previously stated values:

NWF	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 1	158,93	583,44	24,89	42933,3
SIM 2	166,62	1180,01	23,99	41845,95
SIM 3	146,32	511,17	24,86	42933,3
SIM 4	72,16	978,57	47,45	80139,65
SIM 5	114,01	2410,93	47,13	80989,8
SIM 6	16,94	105,454	49,57	84985,15
SIM 7	92,91	2447,94	55,49	90326,65
SIM 8	74,27	1811,32	55,72	89959,9
SIM9	81,13	1171,62	55,98	90539

NWF w/ time priority	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 10	158,93	583,44	24,89	42933,3
SIM 11	166,62	1180,01	23,99	41845,95
SIM 12	146,32	511,17	24,86	42933,3
SIM 13	72,162	978,57	47,45	80139,65
SIM 14	104,17	2440,56	46,89	80696,95
SIM 15	16,94	105,45	49,57	84985,15
SIM 16	92,91	2447,93	55,49	90326,65
SIM 17	82,08	1524,84	56,02	90393,15
SIM 18	81,13	1171,62	55,98	90539,05

NVF	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 19	170,23	637,83	24,01	40964,45
SIM 20	160,83	3309,55	23,29	40271,4
SIM 21	161,92	551,15	21,39	35963,85
SIM 22	95,40	1421,9	45,12	74081,55
SIM 23	89,41	1497,37	40,59	67354,2
SIM 24	69,11	242,59	40,97	68076
SIM 25	76,60	1266,85	48,97	78321,35
SIM 26	87,38	1094,03	46,73	74910,95
SIM 27	85,35	1736,61	47,97	77120,3

CBAA	Average Waiting Time	Average Time In Process	Vehicles Usage %	Distance Traveled by Vehicles
SIM 28	166,99	619,91	22,81	38817,45
SIM 29	160,81	3309,53	23,09	39847,15
SIM 30	168,08	555,21	21,06	35293,3
SIM 31	110,41	1900,68	39,83	63886,2
SIM 32	93,48	2177,99	38,71	62014,9
SIM 33	79,61	1842,87	39,97	63591,7
SIM 34	85,39	3074,25	46,53	70145
SIM 35	89,94	2909,02	46,56	71344,85
SIM 36	71,21	2129,39	47,2	72160,8

Figure 11: Simulation results for Third Layout Scenario

Results Comparison

For comparing the results previously stated the creation of four types of graphs was done. Each graph compares the performance of the dispatching rules based on one of the four variables: *"Average Waiting Time", "Average Time in Process", "Vehicles Usage Percentage" and "Distance Traveled by Vehicles".*

First layout scenario

In all these four graphs four colors can be distinguished. While blue color represents simulations from one to eight, which test NWF, red represents simulations from nine to sixteen and test NWF with time priority. Pink columns belong to simulations from seventeenth to twenty-fourth, testing NVF. Finally, green color represents simulations from twenty-fifth to thirty-second, testing CBAA.

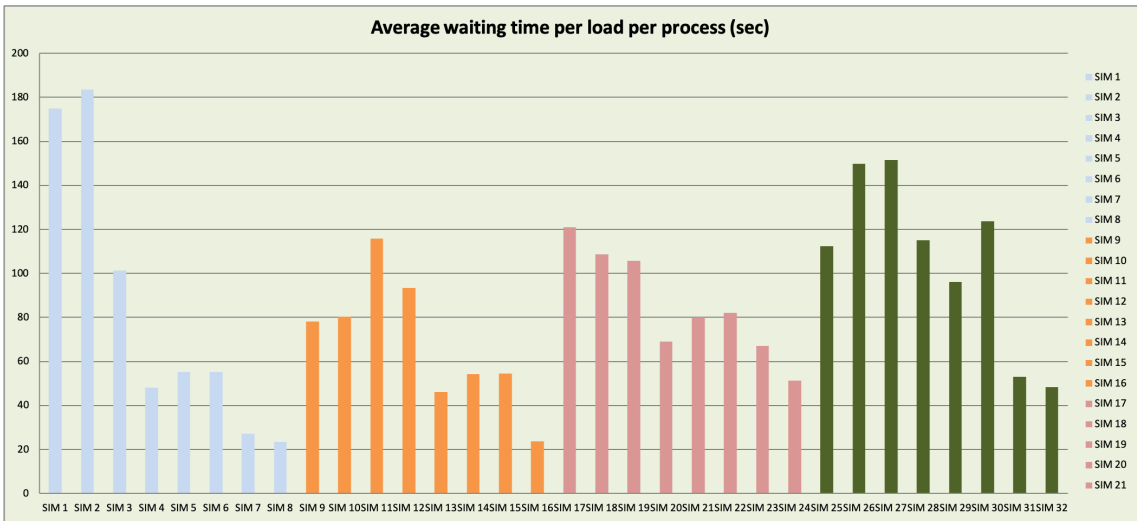


Figure 12: Average waiting time per load per process for first layout scenario

From a first glance at "Average Waiting Time per Load per Process" it can be inferred that the dispatching rule minimizing the most the average waiting time in average among simulations is the NWF with time priority. Analyzing blue columns, simulating NWF, it can be seen how this rule performs really well compared to the others despite of the two first simulations, where loads available at remote stations waited for a long time, which increased the mean average waiting time. NVF, present in the graph

as pink columns, performed better than CBAA but worse than NWF and NWF with time priority. Finally, CBAA was the worst performing dispatching rule in these simulations.

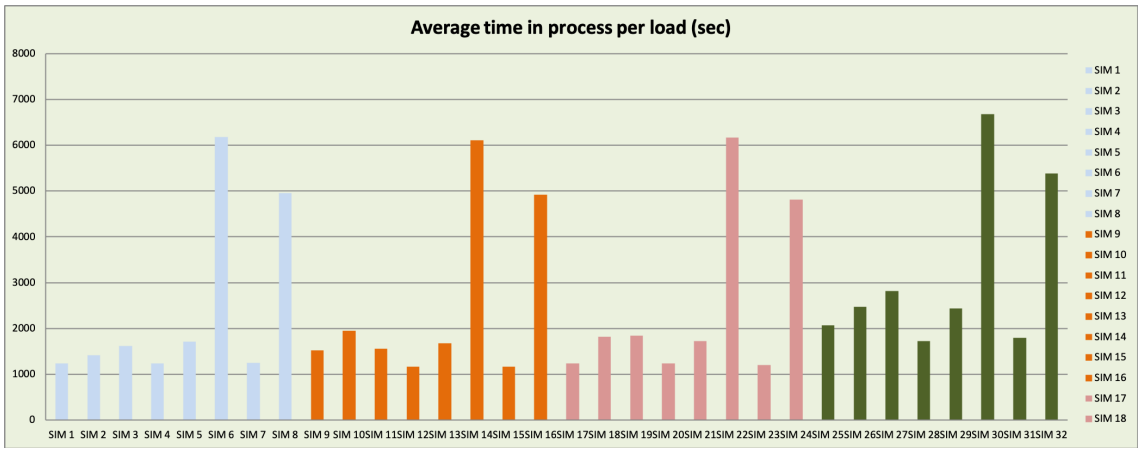


Figure 13: Average time in process per load for first layout scenario

Looking at the graph obtained by comparing the average time it takes for a load to go through the whole process it can be seen how some of the times obtained can be considered, with a 95% of confidence, outliers. This is because at some load creation rate, and number of loads per pallet workers get saturated, which leads to an increase of the average time in process and makes this variable unreliable for that cases. So, for comparing results, these outliers were not taken into account.

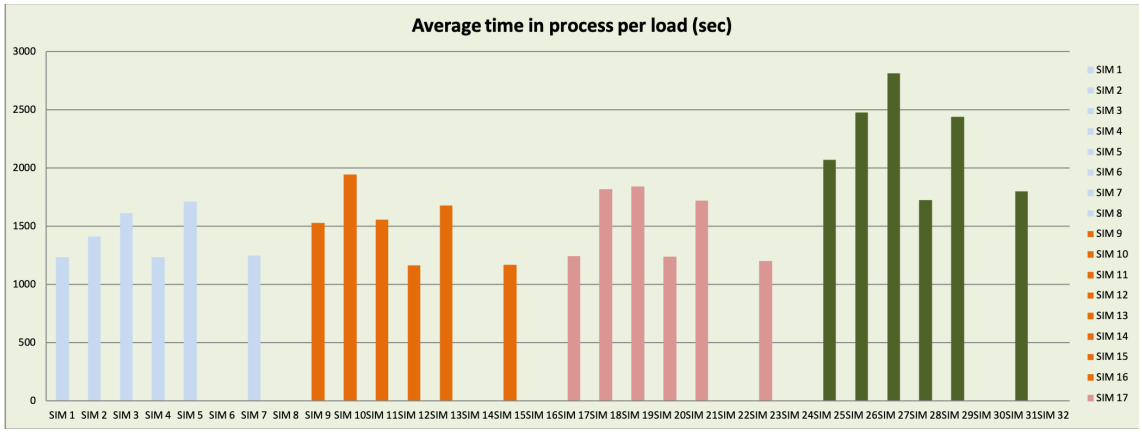


Figure 14: Average time in process per load for first layout scenario

Looking at this new graph, where outliers were removed, it can be seen how NWF, NWF with time priority and NVF have a better performance than CBAA. Analyzing the three first dispatching rules data for these variables, it can be stated with 95% of confidence that these three dispatching rules have a similar performance.

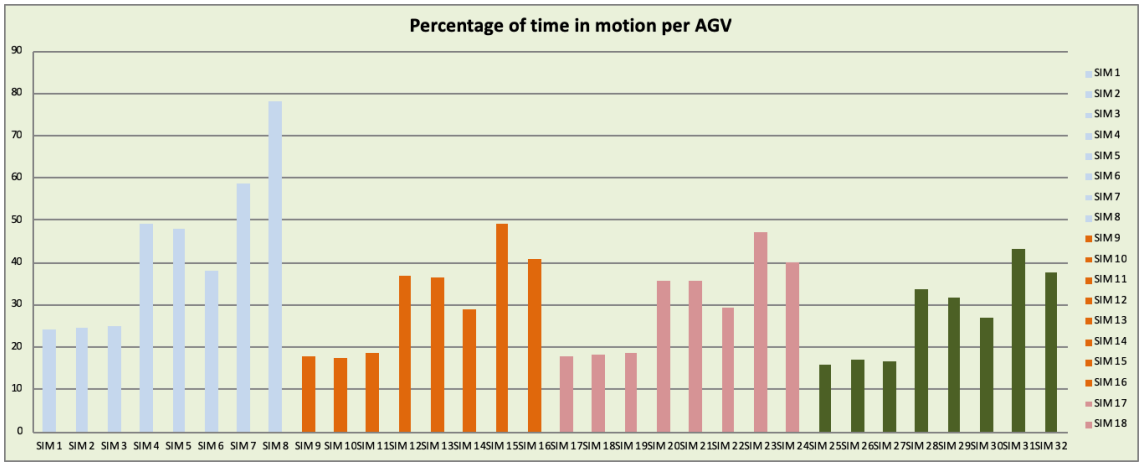


Figure 15: Percentage of time in motion per AGV for first layout scenario

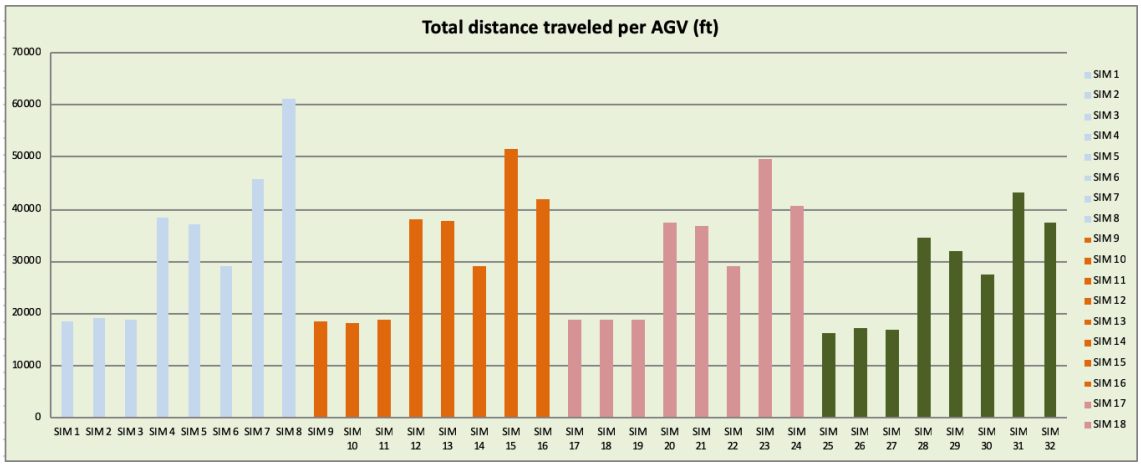


Figure 16: Total distance traveled per AGV for first layout scenario

Average percentage of time in motion per each AGV and the total distance traveled per AGV per simulation are strongly related in this case. This is because AGVs are traveling at the same speed. Even though all four dispatching rules being evaluated have a similar performance, the best dispatching rule in terms of reducing percentage

time in motion and total time distance traveled per each AGV is CBAA, even though in the last simulation it performs worse way than CBAA.

Second layout scenario

The colors presented in the next graphs are the same as the ones in the previous scenario, being blue NWF, orange NWF w/ Time Priority, pink NVF and green CBAA. As previously discussed, the difference between this scenario and the previous one is the addition in this case of a second "process line", which should help with reducing times.

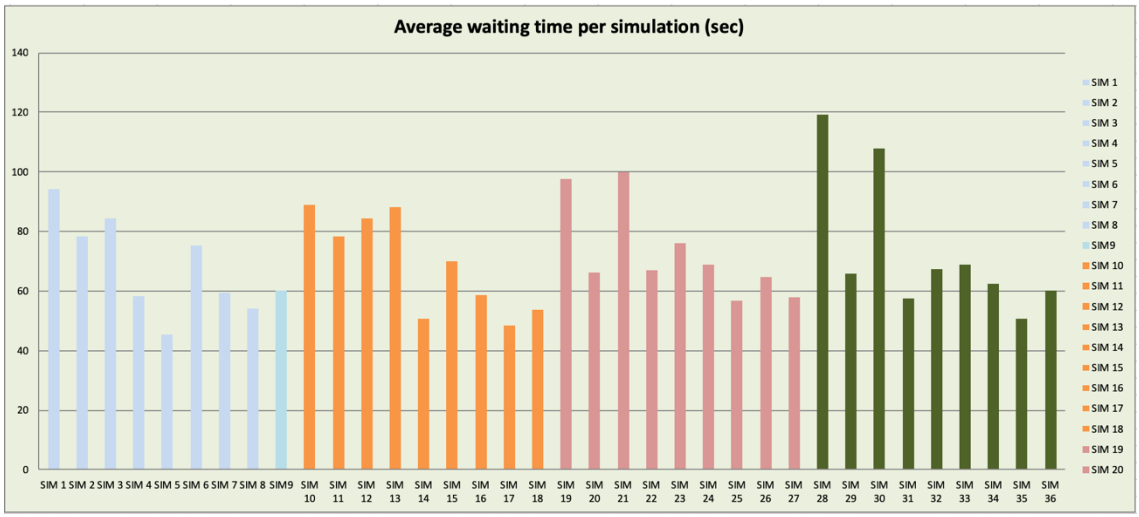


Figure 17: Average waiting time per simulation for second layout scenario

In this scenario we have one more simulation for each dispatching rule. From a first glance, performances of the NWF, NWF w/ Time priority and NVF dispatching rules look better than the performance of the CBAA, which has two well-noticeable outliers. As in the previous scenario, NWF w/ Time priority reduces the waiting time per simulation in the higher values of NWF. This is because of the presence of remote stations. In the next graph, I am going to eliminate simulations one and three to have a clearer view of the overall performance.

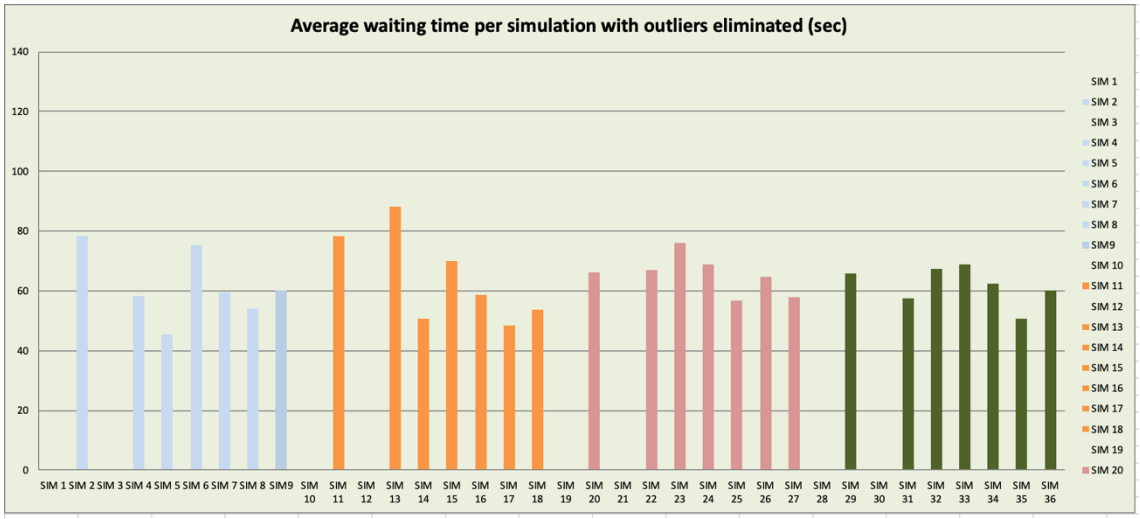


Figure 18: Average waiting time per simulation with outliers eliminated for second layout scenario

If eliminating simulations one and three, as it is done in the figure above, it can be seen that CBAA and NVF work the best for simulation two. For simulation four and five NWF works the best. For simulation six NVF and CBAA perform similarly, and better than NWF and NWF w/ Time Priority. While for simulation seven NVF performs the best, for simulation eight CBAA is the best-working dispatching rule. Finally, for simulation nine, NWF w/ Time Priority performs the best. In the overall, even though in some cases other rules perform better, CBAA has the most equilibrate and best performance. NWF w/ Time Priority most of the times reduces the times achieved by NWF, and in other cases it either remains the same, or makes them worse. This is depending on the threshold value selected.

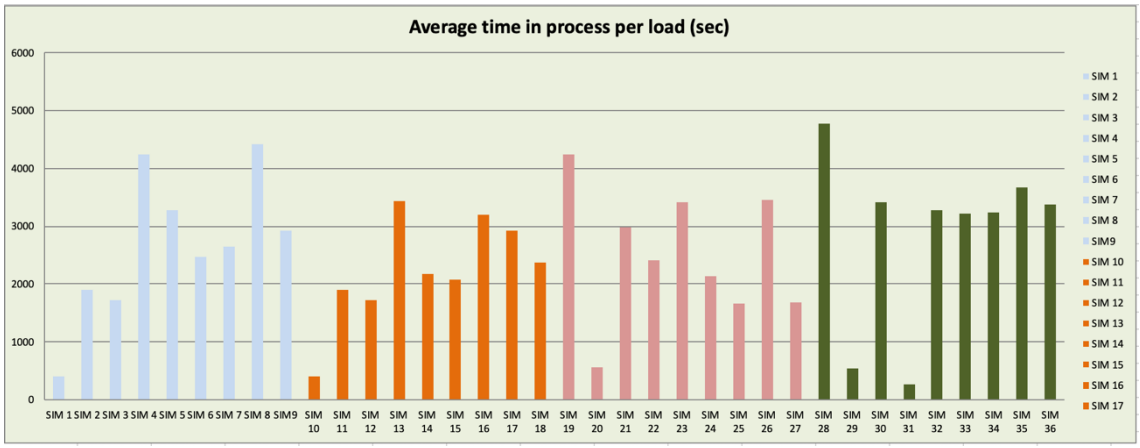


Figure 19: Average time in process per load for second layout scenario

Now, the evaluation of the average time in process per load is going to be performed. Looking at the graph, it can be seen how NWF and NWF w/ Time Priority are the best performing rules for first and third simulations. For simulations two and four CBAA is the best working dispatching rule. For the fifth simulation CBAA and NWF work the best. In simulation six NWF w/ Time Priority and NVF perform the best. In simulations seven and nine NVF is the best dispatching rule and finally, for simulation eight NWF w/ Time is the best one. If comparing NWF w/ Time with NWF it can be seen how in the overall, it improves the performance. The most particular cases in this case are simulations from one to four. The performance of the vehicle initiated dispatching rules is better in simulations one and three, and in simulations two and four the load-initiated dispatching rules perform better.

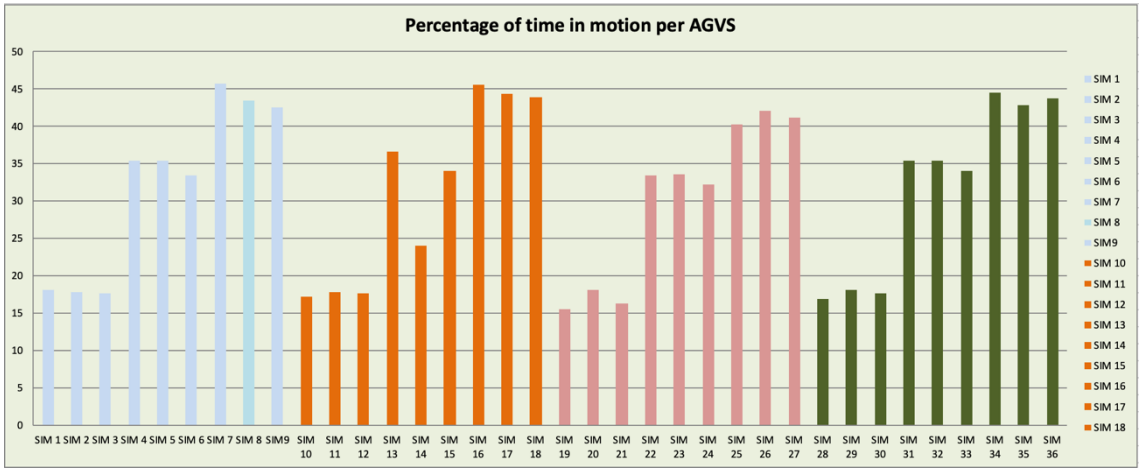


Figure 20: Percentage of time in motion per AGV for second layout scenario

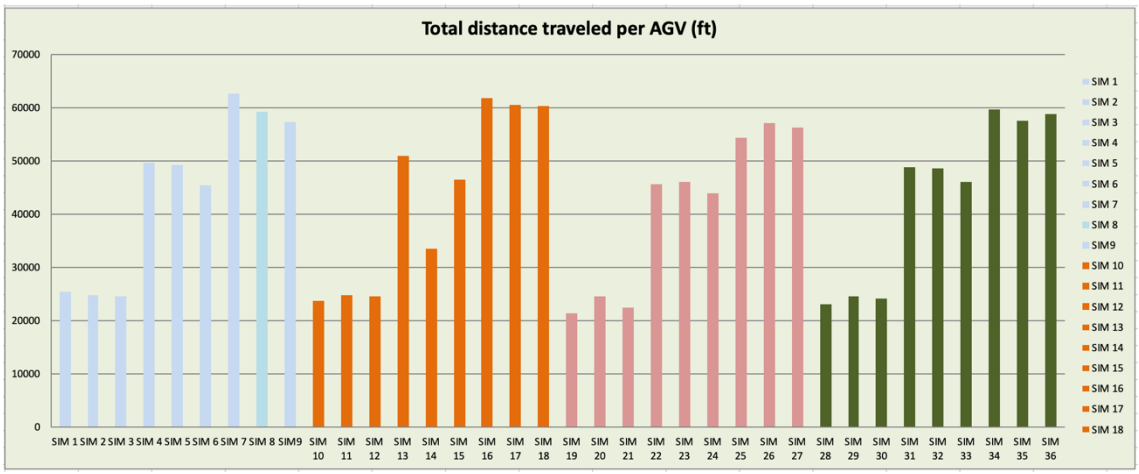


Figure 21: Total distance traveled per AGV for second layout scenario

Percentage of time in motion of the AGVs and total distance traveled can be evaluated together. This is because both AGVs working in the system have the same attributes and characteristics, which evolves in to having both these variables strongly related. For the first three simulations the performance of all four dispatching rules is similar. For simulation number four NVF performs slightly better. For the fifth simulation NWF w/ Time Priority improves both variables being evaluated considerably. For the rest of the simulations the performance is similar, being in all cases NVF the best working rule. From these graphs, it can be seen how the performance between cases does not vary significantly. The only particular case is simulation five, where NWF w/ Time Priority reduces considerably both the variables being evaluated. This can be because of the threshold value set up. If observing the average waiting time graph, this dispatching rule also reduces considerably the time, which means that vehicles were nearer to the stations, which concludes in less distance covered by the vehicles, and less percentage of time in motion.

Third layout scenario

In this third simulation colors are same as before, being blue NWF, orange NWF w/ Time Priority, pink NVF and green CBAA. As previously discussed, this scenario has a loop approach, which will generate particular and different performances compared to the previous scenarios.

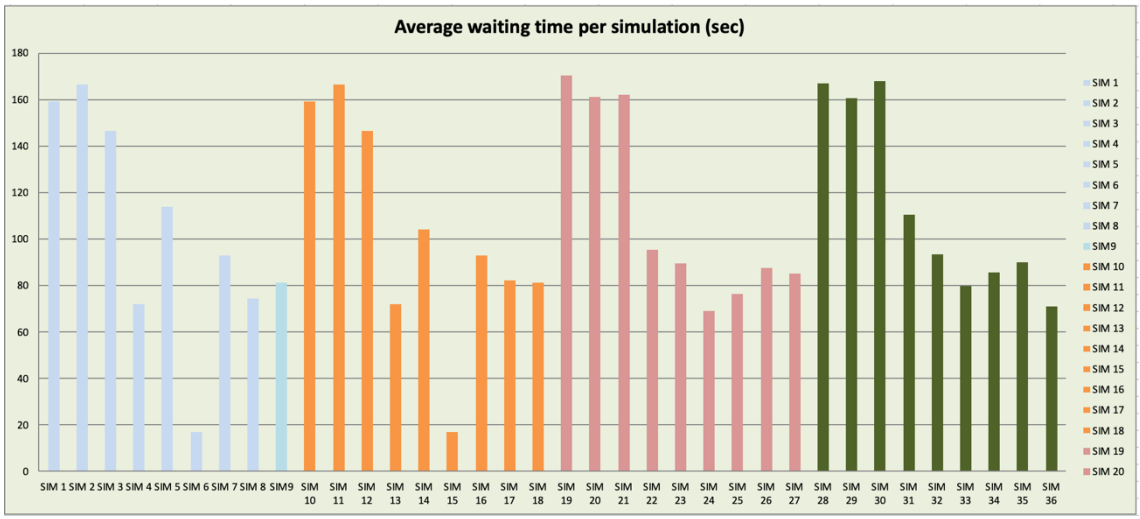


Figure 22: Average waiting time per simulation for third layout scenario

At a first glance, simulation number six draws the attention. It can be seen how NWF and NWF w/ Time Priority strongly improve the average waiting time per simulation in simulation six. While these two dispatching rules are vehicle-initiated, the performance of the load-initiated dispatching rules in this case is way worse. For simulations one, two and three, the performance of all four dispatching rules is similar; being the best performing rules NWF, CBAA and NWF respectively. For simulation four again NWF and NWF w/ Time Priority perform the best. In simulation seven NWF w/ Time Priority improves the performance of NWF but NVF performs better. Although, for simulation eight all dispatching rules perform similarly, the best performing dispatching rule is NWF. Finally, for simulation nine the best performing dispatching rule is CBAA. Overall, the best performance is achieved by NWF w/ Time Priority.

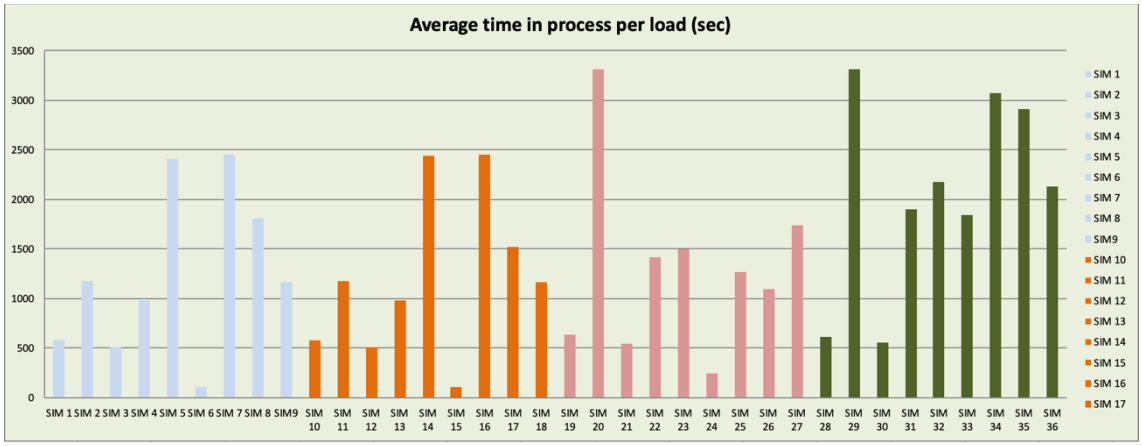


Figure 23: Average time in process per load for third layout scenario

In terms of average time in process per load it can be seen how in the first simulation the times are similar for the four dispatching rules. Even so, in the second simulation differences in performance can be easily noticed. The performance of CBAA and NVF almost triplicates the time value achieved by NWF and NWF w/Time Priority. While, for third simulation results are similar, for the fourth one NVF and CBAA perform considerably worse. For fifth simulation CBAA and NVF perform better than the other two dispatching rules, being NVF the best-working rule. For simulation number six, again the performance of NWF and NWF w/ Time Priority is the best, but, while NVF is not far from performing as these two, CBAA performs considerably worse. For simulation seven and eight NVF performs the best. Finally, for simulation 9 NWF works the best.

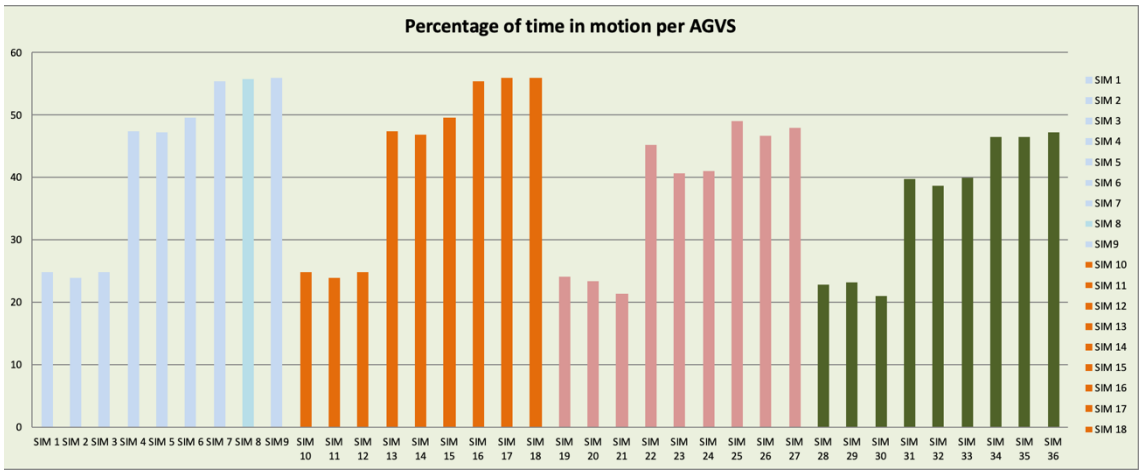


Figure 24: Percentage of time in motion per AGV for third layout scenario

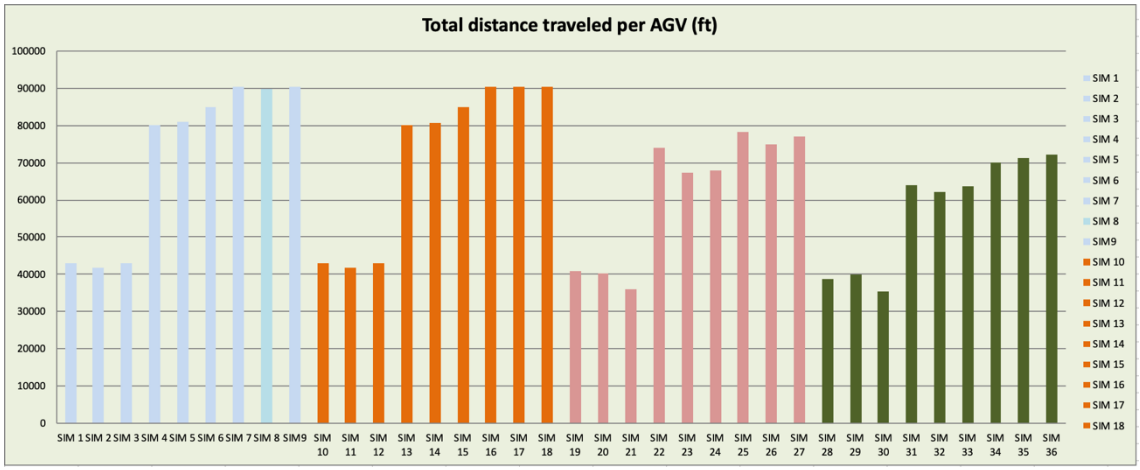


Figure 25: Total distance traveled per AGV for third layout scenario

As in previous scenarios, these two variables will be evaluated together. At a first glance, it can be seen how for first three simulations the performance in terms of these two variables is quite similar. For the next six simulations (from simulation four to simulation nine) the performance of NVF and CBAA is better than the performance for the other two rules. Analyzing deeper, although NVF and CBAA perform similarly, this last reduces the percentage of time in motion and the total distance traveled in AGVs compared with NVF.

Summary

After the previous data comparison is time to draw conclusions. Before starting the comparison one thing should be remarked. When increasing the number of loads per pallet, the average waiting times tend to reduce, as the number of loads generated is less and vehicles have fewer tasks to complete. The opposite happens with the time in process; as workers have more loads to work with time increases.

If looking at the first layout scenario simulations, Nearest Workstation First with Time Priority is the best performing dispatching rule in terms of reducing the average waiting time per load per process. It makes sense that it performs better than its original mater dispatching rule (Nearest Workstation First), as it assigns longest waiting loads, usually located in remote stations where AGVs do not travel as frequently as to others, to a vehicle, while in Nearest Workstation First these loads can be waiting for much longer times. As it has been mentioned along the project, the threshold value assigned should be carefully established for each particular simulation. If it is too low, this dispatching rule will perform worse than the original one and if the value is too high it will perform identically to the original one. Sometimes reducing the average waiting time per load leads to worsen the time in process per load. This can be seen when comparing these two dispatching rules. In simulations where vehicles are more distributed through the entire layout Nearest Workstation First performs as good as NWF with time priority or better. Overall, if setting the goal in reducing average waiting time per load, these two rules perform better than Nearest Vehicle First and Consensus Based Auction Algorithms Applied to Vehicle Scheduling. Also, in some cases workers get saturated, which makes the time in process variable non-reliable.

If optimizing vehicles performance, the best-working dispatching rule is CBAA. Total distance traveled and percentage of time vehicles are used reduces in comparison with the other three dispatching rules. This makes sense, as in the formulation, this dispatching rule minimizes an objective function which has present the unit cost (the inverse of velocity). Although it performs better in terms of vehicles performance, the

poor performance it has in optimizing time-based variables makes it not an option to implement, unless vehicles traveling cost is much more expensive than time cost. The other three dispatching rules have a similar performance in terms of vehicle optimizing.

As mentioned earlier, second layout scenario includes a second "process line". Considering the average waiting time per load, after eliminating two outliers, the Consensus Based Auction Algorithms applied to Vehicle Scheduling have the most equilibrate performance. Again, NWF w/ Time Priority improves the NWF performance if the threshold value is well set up. If that is not the case, the performance can either remain the same or worsen. In this particular case, NWF w/ Time Priority performs better in four cases. All this, setting up the threshold value between four or five times.

If looking at the average time in process, for simulation one, a particular case takes place. Nearest Workstation First and Nearest Workstation with Time Priority have the best performance in terms of this variable, which also coincides with the results of average waiting time per load. For simulation two the same situation takes place, but in this case with Consensus Based Auction Algorithms and Nearest Vehicle First. In this case, the difference is way larger. The major difference takes place in simulation number four, where Nearest Workstation First and Consensus Based Auction Algorithm perform similarly for average waiting time, but in the average time in process the Consensus Based Auction Algorithms performs five times better. One reason for that is that the impact the average waiting time has in the average time in process is not that big. Another reason is that nearer does not mean better. In some cases, as considered in the Consensus Based Auction Algorithms, a driver can be handling another operation before doing the actual one, but, even so, the time it could take this driver to finish the task he is doing and going to the station where the other load is waiting, could be fewer than the time it would take an idle vehicle to go from its actual location to the stations requiring the pick-up. Another interesting case is the one taking place in simulation five, where Nearest Workstation First is the best performing rule in terms of waiting time. The waiting time value for Nearest Workstation First with Time Priority is slightly higher, which could lead anyone to think that this dispatching rule should work better in terms of time in process, but in

this case, Nearest Workstation First with Time Priority performs better in terms of time in process and the difference grows considerably. In this case, taking remote stations into account makes sense (in terms of process time) because, even though they increase the average waiting time, its consideration decreases the average time in process. Putting it briefly, the small increase in the waiting time leads to a considerably higher reduce in time in process.

In facility layout scenario two, if looking at the percentage of time in motion and total distance traveled a particular case can be noticed. It is the case of simulation five. In this case Nearest Workstation First with Time Priority reduces almost a 10% the time in motion and total distance traveled from its best competitor. As the other two simulations realized for the same dispatching rule are in-line with the performance of the other rules, it can be concluded that this reduction is a thing of coincidence.

Another remarkable thing is, in some cases, the importance of selection of the type of dispatching rule, whether it is vehicle-initiated or load-initiated. As explained before, in vehicle-initiated dispatching rules the vehicle looks for jobs to perform, while in load-initiated rules it is the load the responsible for searching for jobs to perform. The better performance of one or another depends on the situation at what loads appear in the system.

As the evaluation of results goes forward, it is noticeable that the waiting times are only a part of the total times in process. Its reduction does not imply the reduction of the average time in process per load.

Finally, particular cases that took place in layout scenario three will be analyzed. The main characteristic of this scenario is its loop approach for the path, and as it will be analyzed this has a huge effect on the dispatching rules performance. Looking at the waiting time data and graphs simulation six draws the attention. Nearest Workstation First and Nearest Workstation First with Time Priority have the same performance, which indicates that the threshold value is not used. For time in process again these two rules

are the best performing, but the other two are not far, which means that the difference in performance is mainly because of the waiting times. Another special case is found in simulation two, where the waiting times performance is similar, or even better for the load-initiated dispatching rules, but in time in process the performances for vehicle-initiated dispatching rules are way better. This is because in this simulation the number of loads generated or appearing (arriving at a station and needing to be picked up) in the lower stations (process lines) section is larger than the number of loads generated in the higher stations (where are situated the trucks entrance, storage areas and the check station).

In the overall, the performance in this case scenario depends on the generation rate and the number of loads per pallet, as these variables change the times of the operators, which directly affects the time in process. For low generation rates (low number of pallets arriving to the system) and a low number of loads per pallet the performance of all dispatching rules is similar. As generation rates grow, the performance for the different dispatching rules changes.

Also, it can be noticed how waiting times for these scenarios do not vary as much as they changed in previous scenarios. This is because the loop approach for the path. This type of path allows fewer shortcuts and makes vehicles work in a more rigid way, with fewer variations.

If evaluating vehicles performance in the overall, two variables should be taken into account. As previously mentioned, these two variables are strongly related due to vehicles having the same velocities and accelerations, which makes distance and percentage of time related. For these two variables load-initiated dispatching rules perform better. In facility scenario one the Consensus Based Auction Algorithms have the best performance. For second scenario Nearest Vehicle First performs the best, and finally, for third scenario, the best-performing dispatching rules are Nearest Vehicle First and the Consensus Based Auction Algorithms.

Putting it briefly, vehicle-initiated dispatching rules work better in terms of minimizing times and load-initiated dispatching rules perform better in terms of minimizing vehicles usage and vehicles distance covered. The more minimized the times are, the more percentage of use and distance covered by vehicles present in the system.

Conclusion and final thoughts

As it has been evaluated through this Project, Nearest Workstation First and Nearest Workstation First perform the best for low generation rates (large time between each load generated). For medium load generation rates CBAA starts performing better, but Nearest Workstation First with Time Priority continues being the most equilibrate-best- performing rule. Finally, for high load generation rates, Nearest Workstation First Works the best. In the overall, vehicle-initiated dispatching rules work better. Also, it is worthy to state that, in some cases, Nearest Vehicle First or the Consensus Based Auction Algorithms perform way better than any other rule. This means that, if consequently adapting each dispatching rule for each scenario possible, results with the Consensus Based Auction Algorithms can be the best ones.

Also, this great performance achieved by the vehicle-initiated dispatching rules has something to do with how vehicles were set up. Vehicles velocities did not vary between simulations, which made the distance the main variable to optimize.

Workers saturation is another aspect to take into account. As it has been discussed through the development of the project, the number of loads per pallet needs to be carefully set up in order to not have operators saturated. Also, number of loads a worker can handle at a time needs to be constantly revised. In other cases, depending on the operator and its specialization times should be readjusted. Saturation leads to a huge increase even if it is only happening in a small number of cases.

In terms of type of layouts, as it is obvious, having more process lines gives the factory the ability to handle more loads without saturation. Also, every investment made in terms of adding process lines has to be deeply revised. Adding more process lines adds also more operators and depending on the volume of work the factory needs to handle it can be damaging in terms of Money.

In first two scenarios the path approach was the same; a loop approach with a large number of shortcuts. For the third scenario, a loop approach itself was implemented. When implementing it, the differences between performances of the dispatching rules were reduced, compared with the other two scenarios.

As previously stated, minimization of times is inversely proportional to minimization of distances. Knowing that, the company has to decide between affording the cost of more vehicle utilization or affording the cost of time delay and wait time.

Finally, simulation should be seen and used as a help tool, but always leaving final decisions to humans. Along this project lots of approaches and assumptions have been taken, which makes simulations differ from reality. In real world everything can happen, from workers not performing their job at scheduled times, to a worker not coming one day to job. Other situations related, for example, with AGVs can happen; AGVs can fail, or they could need more time to fuel, or velocity cannot always be the same. Even so, simulating real-world facilities in programs such as the one I have worked with (AutoMod) can really help with processes such as the design of the layout of a plant or the optimization of internal transport system, as it was done in this project. Simulations as the ones performed in this project can make the difference when optimizing any type of facility. These simulations can give a good-idea for the human making decisions and the more information, tools and simulations someone has before making a decision, the better decision this person will make.

Appendices

Appendix A. Code for first layout scenario

```
begin P_TruckIn arriving procedure
move into Q_TruckIn
set A_Time to ac
set A_Waitingtime to 0
use R_Operator3 for uniform 75,10 sec
set A_Time1 to ac
move into AGV.cptruckin
set A_Waitingtime to ac - A_Time1

send to P_StoreIn
end

begin P_StoreIn arriving procedure

travel to AGV.cpstoreinIN
/*use R_Operator4 for uniform 10,2 sec*/
move into Q_StoreIn
use R_Operator4 for uniform 60,5 sec
send to P_StoreInOut
end

begin P_StoreInOut arriving procedure
use R_Operator4 for uniform 10,2 sec
set A_Time1 to ac
move into AGV.cpstoreinOUT
set A_Waitingtime to A_Waitingtime + ac - A_Time1
send to P_Depalletization
```

```
end

begin P_Depalletization arriving procedure
travel AGV.cpprocessin
move into Q_Depalletization
use R_Operator7 for uniform 35,3
use R_Operator1 for uniform 270,15 sec
clone 5 loads to P_EnterAssembly new load type L_Coil2
/*use R_Operator1 for uniform 140,3 sec*/

end

begin P_EnterAssembly arriving procedure
set type to L_Coil2
move into Q_ProcessIn
move into conv.sta1
send to P_CompleteAssembly
end

begin P_CompleteAssembly arriving procedure
travel to conv.sta2
inc V_Palletization by 1
use R_Operator2 for uniform 40,4 sec
move into Q_ProcessOut
/*if V_Palletization = 80 then
begin creo que lo que no funciona es el V_Palletization, se puede ver poniendolo en la
pantalla */
/* If V_Palletization is equal to 80... and set V_Palletization to 0 */
/*+ clone 1 loads to P_Palletization new load type L_Coil
set V_Palletization to 0
end*/
```

```
/*send to P_Palletization*/
if V_Palletization = 5 then
begin
clone 1 loads to P_Palletization new load type L_Coil3
set V_Palletization to 0
use R_Operator8 for uniform 30,5 sec
end
/*send to P_Palletization*/

end

begin P_Palletization arriving procedure

set type to L_Coil3
move into Q_Palletization
set A_Time1 to ac
move into AGV.cpprocessout
set A_Waitingtime to A_Waitingtime + ac - A_Time1
send to P_StoreOut
end

begin P_StoreOut arriving procedure

travel to AGV.cpstoreoutIN

use R_Operator5 for uniform 10,2 sec
move into Q_StoreOut
send to P_StoreOutOut
end

begin P_StoreOutOut arriving procedure
```

```
use R_Operator5 for uniform 10,2 sec
set A_Time1 to ac
move into AGV.cpstoreoutOUT
set A_Waitingtime to A_Waitingtime + ac - A_Time1
send to P_TruckOut
end

begin P_TruckOut arriving procedure
travel to AGV.cptruckout
move into Q_TruckOut
use R_Operator6 for uniform 75,7 sec
inc V_Numloads by 1
set A_Time to ac - A_Time
print this load, "Time in system " A_Time to message
set A_Timecumulative to A_Timecumulative + A_Time
print this load, "Time cumulative " A_Timecumulative to message
set A_Averagetime to A_Timecumulative / V_Numloads
print this load, "Average time " A_Averagetime to message
print this load, "Number of loads processed " V_Numloads to message
print this load, "Waiting time sum is " A_Waitingtime to message
set A_Waitingtimecumulative to A_Waitingtimecumulative + A_Waitingtime
set A_Averagewaitingtime to A_Waitingtimecumulative / V_Numloads
print this load, "Average waiting time is " A_Averagewaitingtime to message
send to die
end
```

Appendix B. Code for second layout scenario.

```
begin P_TruckIn arriving procedure
        move into Q_TruckIn
        set A_Time to ac
```

```
        set A_Waitingtime to 0
        use R_Operator3 for uniform 75,10 sec
        set A_Time1 to ac
        move into AGV.cptruckin
        send to P_StoreIn
end

begin P_StoreIn arriving procedure
        travel to AGV.cpstoreinIN
        /*use R_Operator4 for uniform 10,2 sec*/
        move into Q_StoreIn
        use R_Operator4 for uniform 60,5 sec
        send to P_StoreInOut
end

begin P_StoreInOut arriving procedure
        use R_Operator4 for uniform 10,2 sec
        set A_Time1 to ac
        move into AGV.cpstoreinOUT
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to oneof(50:P_Depalletization,50:P_Depalletization2)
end

begin P_Depalletization arriving procedure
        travel AGV.cpprocessin
        move into Q_Depalletization
        use R_Operator7 for uniform 35,3
        use R_Operator1 for uniform 270,15 sec
        clone 10 loads to P_EnterAssembly new load type L_Coil2
        /*use R_Operator1 for uniform 140,3 sec*/
end
```

```
begin P_Depalletization2 arriving procedure
        travel AGV.cpprocessin2
        move into Q_Depalletization2
        use R_Operator72 for uniform 35,3
        use R_Operator12 for uniform 270,15 sec
        clone 5 loads to P_EnterAssembly2 new load type L_Coil2
        /*use R_Operator1 for uniform 140,3 sec*/
end

begin P_EnterAssembly arriving procedure
        set type to L_Coil2
        move into Q_ProcessIn
        move into conv.sta1
        send to P_CompleteAssembly
end

begin P_EnterAssembly2 arriving procedure
        set type to L_Coil2
        move into Q_ProcessIn2
        move into conv.sta3
        send to P_CompleteAssembly2
end

begin P_CompleteAssembly arriving procedure
        travel to conv.sta2
        inc V_Palletization by 1
        use R_Operator2 for uniform 40,4 sec
        move into Q_ProcessOut
        /*if V_Palletization = 80 then
```

```
        begin creo que lo que no funciona es el V_Palletization, se puede ver poniendolo
en la pantalla */
        /* If V_Palletization is equal to 80... and set V_Palletization to 0 */
/*+      clone 1 loads to P_Palletization new load type L_Coil
                set V_Palletization to 0
        end*/
        /*send to P_Palletization*/
        if V_Palletization = 10 then
                begin
                clone 1 loads to P_Palletization new load type L_Coil3
                set V_Palletization to 0
                use R_Operator8 for uniform 30,5 sec
                end
        /*send to P_Palletization*/

end

begin P_CompleteAssembly2 arriving procedure
        travel to conv.sta4
        inc V_Palletization2 by 1
        use R_Operator22 for uniform 40,4 sec
        move into Q_ProcessOut2
                if V_Palletization2 = 5 then
                begin
                clone 1 loads to P_Palletization2 new load type L_Coil3
                set V_Palletization2 to 0
                use R_Operator82 for uniform 30,5 sec
                end
end

begin P_Palletization arriving procedure
```

```
        set type to L_Coil3
        move into Q_Palletization
                set A_Time1 to ac
        move into AGV.cpprocessout
                set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to P_StoreOut
end

begin P_Palletization2 arriving procedure
        set type to L_Coil3
        move into Q_Palletization2
                set A_Time1 to ac
        move into AGV.cpprocessout2
                set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to P_StoreOut
end

begin P_StoreOut arriving procedure
        travel to AGV.cpstoreoutIN
        use R_Operator5 for uniform 10,2 sec
        move into Q_StoreOut
        send to P_StoreOutOut
end

begin P_StoreOutOut arriving procedure
        use R_Operator5 for uniform 10,2 sec
                set A_Time1 to ac
        move into AGV.cpstoreoutOUT
                set A_Waitingtime to A_Waitingtime + ac - A_Time1

        send to P_TruckOut
```

```
end

begin P_TruckOut arriving procedure
        travel to AGV.cptruckout
        move into Q_TruckOut
        use R_Operator6 for uniform 75,7 sec
        inc V_Numloads by 1
        set A_Time to ac - A_Time
        print this load, "Time in system " A_Time to message
        set A_Timecumulative to A_Timecumulative + A_Time
        print this load, "Time cumulative " A_Timecumulative to message
        set A_Averagetime to A_Timecumulative / V_Numloads
        print this load, "Average time " A_Averagetime to message
        print this load, "Number of loads processed " V_Numloads to message
        if A_Waitingtime = 0 then begin
                print this load, "Waiting time 1 is 0 and it has to be " A_Waitingtime to
message
                print this load, "Average waiting time has to be the same as before and it
is " A_Averagewaitingtime to message
                send to die
                end
        else
                print this load, "Waiting time sum is " A_Waitingtime to message
                set A_Waitingtimecumulative to A_Waitingtimecumulative +
A_Waitingtime
                set A_Averagewaitingtime to A_Waitingtimecumulative / V_Numloads
                print this load, "Average waiting time is " A_Averagewaitingtime to
message
                send to die

end
```

Appendix C. Code for third layout scenario.

```
begin P_TruckIn arriving procedure
        move into Q_TruckIn
        set A_Time to ac
        set A_Waitingtime to 0
        use R_Operator3 for uniform 75,10 sec
        set A_Time1 to ac
        move into AGV.cptruckin
        send to P_Check
end

begin P_Check arriving procedure
        travel to AGV.cpcheckIN
        move into Q_Check
        use R_Operator9 for uniform 20,3 sec
        send to oneof(95:P_StoreIn,5:P_Reject)
end

begin P_Reject arriving procedure
        send to die
end
begin P_StoreIn arriving procedure
        set A_Time1 to ac
        move into AGV.cpcheckOUT
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        travel to AGV.cpstoreinIN
        move into Q_StoreIn
        use R_Operator4 for uniform 60,5 sec
        send to P_StoreInOut
```

```
end

begin P_StoreInOut arriving procedure
        use R_Operator4 for uniform 10,2 sec
        set A_Time1 to ac
        move into AGV.cpstoreinOUT
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to P_Depalletization1
end

begin P_Depalletization1 arriving procedure
        travel AGV.cpdepalletizationin
        move into Q_Depalletization
        use R_Operator71 for uniform 17.5,1.5 sec
        use R_Operator72 for uniform 17.5,1.5 sec
/*      clone 10 loads to P_TraveltoProcess new load type L_Coil2 */
        /*use R_Operator11 for uniform 140,3 sec*/
        send to P_TraveltoProcess
end

begin P_TraveltoProcess arriving procedure
        send                                                    to
oneof(33:P_EnterAssembly1,33:P_EnterAssembly2,33:P_EnterAssembly3)
end

begin P_EnterAssembly1 arriving procedure
/*      set type to L_Coil2  */
        move into AGV.cpdepalletizationout
        travel to AGV.cpprocessin1
        move into Q_ProcessIn1
        use R_Operator11 for uniform 270,15 sec
```

```
        clone 3 loads to P_ProcessClonation1 new load type L_Coil2
end

begin P_EnterAssembly2 arriving procedure
/*      set type to L_Coil2  */
        move into AGV.cpdepalletizationout
        travel to AGV.cpprocessin2
        move into Q_ProcessIn2
        use R_Operator12 for uniform 270,15 sec
        clone 4 loads to P_ProcessClonation2 new load type L_Coil2
end

begin P_EnterAssembly3 arriving procedure
/*      set type to L_Coil2  */
        move into AGV.cpdepalletizationout
        travel to AGV.cpprocessin3
        move into Q_ProcessIn3
        use R_Operator13 for uniform 270,15 sec
        clone 5 loads to P_ProcessClonation3 new load type L_Coil2
end

begin P_ProcessClonation1 arriving procedure
        set type to L_Coil2
        set A_Time1 to ac
        move into conv.sta21
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to P_CompleteAssembly1
end

begin P_ProcessClonation2 arriving procedure
        set type to L_Coil2
```

```
        set A_Time1 to ac
        move into conv.sta31
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to P_CompleteAssembly2
end

begin P_ProcessClonation3 arriving procedure
        set type to L_Coil2
        set A_Time1 to ac
        move into conv.sta41
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to P_CompleteAssembly3
end

begin P_CompleteAssembly1 arriving procedure
        travel to conv.sta22
        inc V_Palletization1 by 1
        use R_Operator21 for uniform 40,4 sec
        move into Q_ProcessOut1

        if V_Palletization1 = 3 then
                begin
                clone 1 loads to P_Palletization1 new load type L_Coil3
                set V_Palletization1 to 0
        /*      use R_Operator81 for uniform 30,5 sec*/
                end


end

begin P_CompleteAssembly2 arriving procedure
```

```
        travel to conv.sta32
        inc V_Palletization2 by 1
        use R_Operator22 for uniform 40,4 sec
        move into Q_ProcessOut2

        if V_Palletization2 = 4 then
                begin
                clone 1 loads to P_Palletization2 new load type L_Coil3
                set V_Palletization2 to 0
        /*      use R_Operator81 for uniform 30,5 sec*/
                end


end

begin P_CompleteAssembly3 arriving procedure
        travel to conv.sta42
        inc V_Palletization3 by 1
        use R_Operator23 for uniform 40,4 sec
        move into Q_ProcessOut3

        if V_Palletization3 = 5 then
                begin
                clone 1 loads to P_Palletization3 new load type L_Coil3
                set V_Palletization3 to 0
        /*      use R_Operator81 for uniform 30,5 sec*/
                end


end
```

```
begin P_Palletization1 arriving procedure
        set type to L_Coil3
        set A_Time1 to ac
        move into AGV.cpprocessout1
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        travel to AGV.cppalletizationin
        move into Q_Palletization
        use R_Operator81 for uniform 15,2.5 sec
        use R_Operator82 for uniform 15,2.5 sec
        send to P_StoreOut
end

begin P_Palletization2 arriving procedure
        set type to L_Coil3
        set A_Time1 to ac
        move into AGV.cpprocessout2
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        travel to AGV.cppalletizationin
        move into Q_Palletization
        use R_Operator81 for uniform 30,5 sec
        send to P_StoreOut
end

begin P_Palletization3 arriving procedure
        set type to L_Coil3
        set A_Time1 to ac
        move into AGV.cpprocessout3
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        travel to AGV.cppalletizationin
        move into Q_Palletization
        use R_Operator81 for uniform 30,5 sec
```

```
        send to P_StoreOut
end

begin P_StoreOut arriving procedure
        set A_Time1 to ac
        move into AGV.cppalletizationout
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        travel to AGV.cpstoreoutIN
        move into Q_StoreOut
        use R_Operator5 for uniform 10,2 sec
/*      move into Q_StoreOut*/
        send to P_StoreOutOut
end



begin P_StoreOutOut arriving procedure
        use R_Operator5 for uniform 10,2 sec
        set A_Time1 to ac
        move into AGV.cpstoreoutOUT
        set A_Waitingtime to A_Waitingtime + ac - A_Time1
        send to P_TruckOut
end

begin P_TruckOut arriving procedure
        travel to AGV.cptruckout
        move into Q_TruckOut
        use R_Operator6 for uniform 75,7 sec
        inc V_Numloads by 1
        set A_Time to ac - A_Time
        print this load, "Time in system " A_Time to message
```

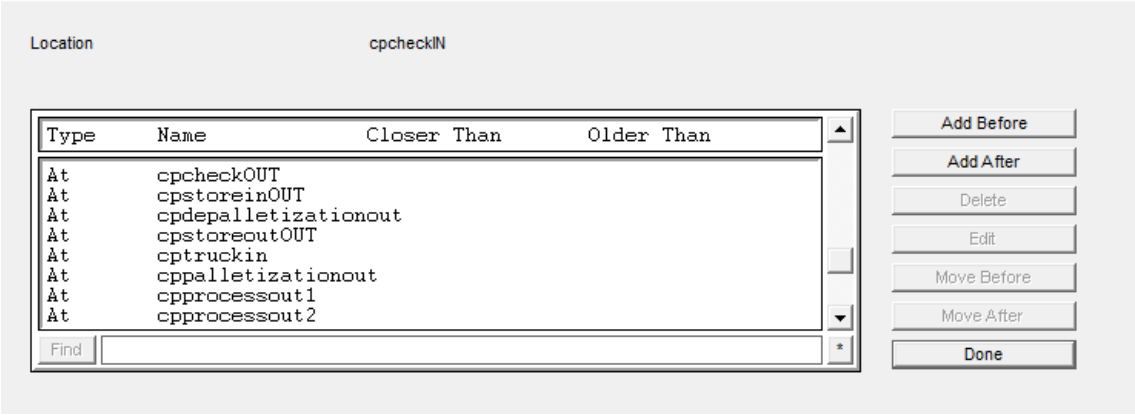
```
        set A_Timecumulative to A_Timecumulative + A_Time
        print this load, "Time cumulative " A_Timecumulative to message
        set A_Averagetime to A_Timecumulative / V_Numloads
        print this load, "Average time " A_Averagetime to message
        print this load, "Number of loads processed " V_Numloads to message
        if A_Waitingtime = 0 then begin
                print this load, "Waiting time 1 is 0 and it has to be " A_Waitingtime to
message
                print this load, "Average waiting time has to be the same as before and it
is " A_Averagewaitingtime to message
                send to die
                end
        else
                print this load, "Waiting time sum is " A_Waitingtime to message
                set A_Waitingtimecumulative to A_Waitingtimecumulative +
A_Waitingtime
                set A_Averagewaitingtime to A_Waitingtimecumulative / V_Numloads
                print this load, "Average waiting time is " A_Averagewaitingtime to
message
                send to die
end



begin AGV load available function
        set V_loadlocation = loadLoc
                for each V_Vehicle in AGV vehicles do begin
                        inc i by 1
                        if V_Vehicle number of loads = 0 then
                                begin
                                set V_distance1 to V_Vehicle path distance to loadLoc
```

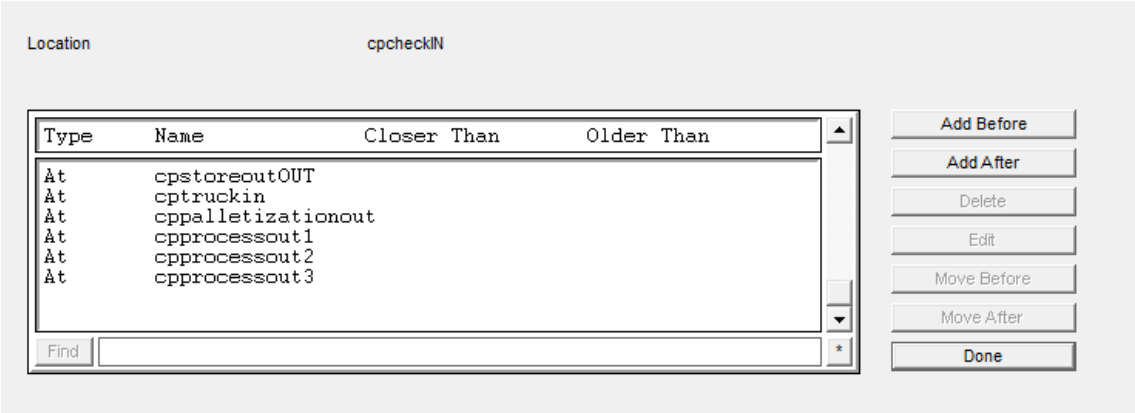
```
                print "The distance from unloaded vehicle is " V_distance1
to message
                set V_bid2 to (V_distance1)/3
                print "The bid for unloaded vehicle is " V_bid2 to message
                if V_bid2 < V_bid1 then
                        begin
                                set V_bid1 to V_bid2
                                set j to i
                                print "The new vehicle selected through first
bidding is " i to message
                        end
                end
        if V_Vehicle number of loads = 1 then
                begin
                set V_distance1 to V_Vehicle path distance to loadLoc
                print "The distance from loaded vehicle is " V_distance1 to
message
                set V_distance2 to V_Vehicle path distance to V_Vehicle
current schedjob location
                print "The distance from loaded vehicle to performing job
is " V_distance2 to message
                set V_arrivaltime to (V_distance1)/3
                set V_timetoload to (V_distance2)/3
                set V_bid2 to V_arrivaltime + V_timetoload
                print "The bid for loaded vehicle is " V_bid2 to message
                if V_bid2 < V_bid1 then
                        begin
                                set V_bid1 to V_bid2
                                set j to i
                                print "The new vehicle selected through
second bidding is " i to message
```

```
                                        end
                                    end
                            end
                    if j = 1 or j = 2 then /*ensure it is not 0*/
                            begin
                                    claim theLoad for AGV.AGVI(j)
                                    print "The final vehicle selected is vehicle " j to message
                            end
                    set i to 0
                    set j to 0
                    set V_bid1 to 9999
                    set V_bid2 to 9999
                    return 0
end
```

Appendix D. Work List sample for Nearest Workstation First

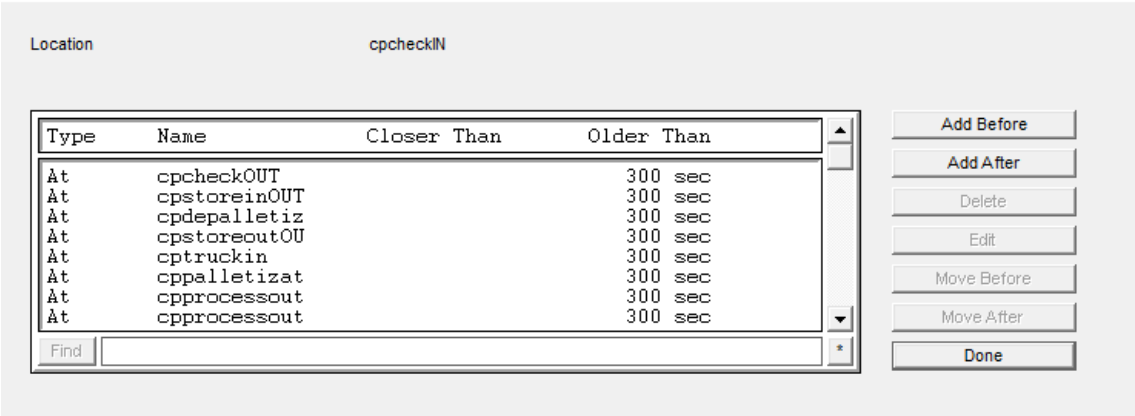
Attached below are some pictures showing how Work Lists are. This Work List is the one for third layout scenario for the station cpcheckin.

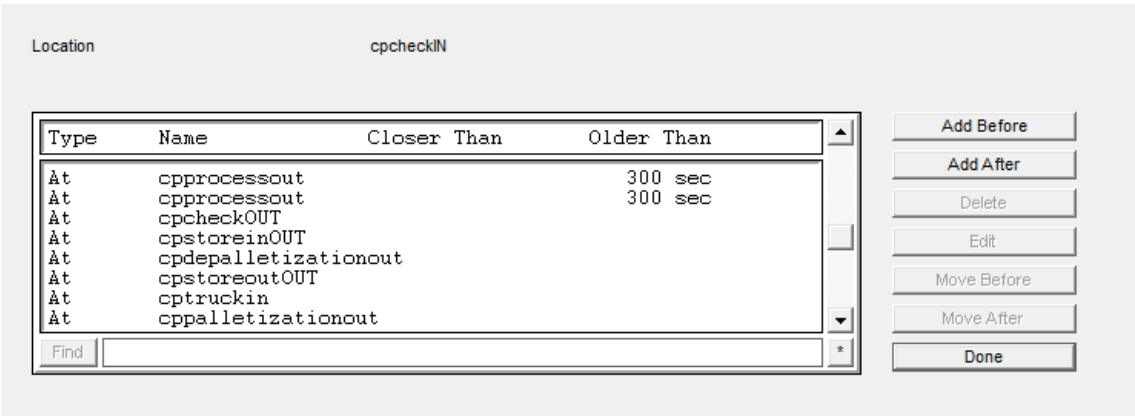


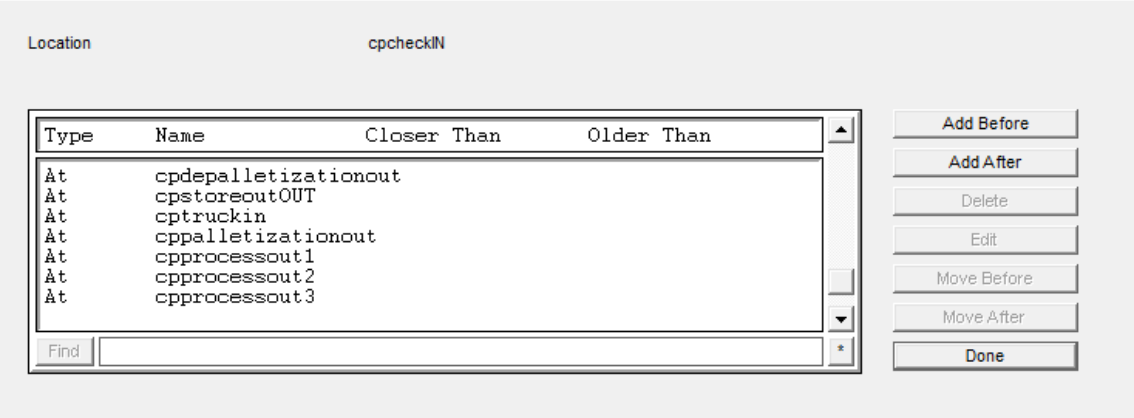


Appendix E: Work List sample for Nearest Workstation First with Time Priority

Attached below are some pictures showing how Work Lists are. This Work List is the one for third layout scenario for the station cpcheckin.







Appendix F. Nearest Vehicle First code

```
begin AGV load available function
set V_distance1 to 0
set V_loadlocation = loadLoc
for each V_Vehicle in AGV vehicles do begin
inc i by 1
if V_Vehicle number of loads = 0 then
begin
set V_distance1 to V_Vehicle path distance to loadLoc
print "The distance from unloaded vehicle is " V_distance1 to message
if V_distance1 < V_distance2 then
begin
set V_distance2 to V_distance1
set j = i
end
end
end
if j = 1 or j = 2 then /*ensure it is not 0*/
begin
claim theLoad for AGV.AGVI(j)
print "The final vehicle selected is vehicle " j to message
```

```
end
set i to 0
set j to 0
set V_distance2 to 9999999
return 0
end
```

Appendix G. Consensus Based Auction Algorithms applied to vehicle scheduling code

```
begin AGV load available function
set V_loadlocation = loadLoc
for each V_Vehicle in AGV vehicles do begin
inc i by 1
if V_Vehicle number of loads = 0 then
begin
set V_distance1 to V_Vehicle path distance to loadLoc
print "The distance from unloaded vehicle is " V_distance1 to message
set V_bid2 to (V_distance1)/3
print "The bid for unloaded vehicle is " V_bid2 to message
if V_bid2 < V_bid1 then
begin
set V_bid1 to V_bid2
set j to i
print "The new vehicle selected through first bidding is " i to message
end
end
if V_Vehicle number of loads = 1 then
begin
set V_distance1 to V_Vehicle path distance to loadLoc
print "The distance from loaded vehicle is " V_distance1 to message
```

```
set V_distance2 to V_Vehicle path distance to V_Vehicle current schedjob location
print "The distance from loaded vehicle to performing job is " V_distance2 to message
set V_distance3 to loadLoc path distance to V_Vehicle current schedjob location
print "The distance from loaded vehicle to performing job is " V_distance3 to message
set V_arrivaltime to (V_distance1)/3
set V_timetoload to (V_distance2 + V_distance3)/3
set V_bid2 to V_arrivaltime + V_timetoload
print "The bid for loaded vehicle is " V_bid2 to message
if V_bid2 < V_bid1 then
begin
set V_bid1 to V_bid2
set j to i
print "The new vehicle selected through second bidding is " i to message
end
end
end
if j = 1 or j = 2 then /*ensure it is not 0*/
begin
claim theLoad for AGV.AGVI(j)
print "The final vehicle selected is vehicle " j to message
end
set i to 0
set j to 0
set V_bid1 to 9999
set V_bid2 to 9999
return 0
end
```

References

- *[HAFI13]* "Intelligent Combinatorial Auctions of Decentralized Task Assignment for AGV With Multiple Loading Capacity". Muhammad Hafidz Fazli bin Md Fauadi, Saifudin Hafiz Yahaya, Tomohiro Murata.
- *[HERA19]* Juan Marcos de las Heras provided the information of the real-world factory.
- *[TANC94]* *"Material Flow Systems in Manufacturing". J.M Tanchoco.*