Research School for Operations Management and Logistics

Simulation of a multi-agent system for autonomous trailer docking

Berry Gerrits, Martijn Mes, Peter Schuur

Beta Working Paper series 503

BETA publicatie	WP 503 (working paper)
ISBN	
ISSN	
NUR	804
Eindhoven	April 2016

Simulation of a Multi-Agent System for Autonomous Trailer Docking

Berry Gerrits, Martijn Mes, Peter Schuur
Department of Industrial Engineering and Business Information Systems
University of Twente, The Netherlands

April 2016

ABSTRACT

This paper presents a simulation model of a generic automated planning and control system for the pick-up and docking of semi-trailers by means of autonomous Yard Tractors (YTs) in a collision- and conflict free environment. To support the planning and control of the YTs, we propose a Multi-Agent System (MAS). We illustrate our approach using a case study at a Dutch logistics service provider. To evaluate the proposed system, we design an agent-based simulation model, which is setup in a similar way as the MAS. We conclude with the implementation and validation of the simulation model and present some results.

1 INTRODUCTION

The past few years have witnessed an increased interest within the logistics sector for automated driving. For logistics service providers it yields an interesting business opportunity to decrease waiting time and maneuvering time at DCs and thus to overcome problems with the drivers tachograph as the driver has to comply with European Law regarding driving times and rest periods. For a truck unloading at a distribution center, there are various ways of implementing automated driving. We distinguish between two options once the truck arrives: (i) the truck driver is assisted using an automated driving system or (ii) the truck and semi-trailer are decoupled and an autonomous vehicle, a so called yard tractor (YT), takes the semi-trailer to the dock. In this paper, we focus on the latter. Our aim is to design (i) a stable and robust planning and control system and (ii) a discrete-event simulation model to evaluate this system.

For the planning and control of transportation resources, in this case the YT, we could resort to operations research (OR) based global optimization methods. However, as stated by MES et al. (2007), these methods might be less suitable for real-time decision making in stochastic and dynamic environments, since they typically (i) require a lot of information in advance, (ii) are sensitive to information updates, (iii) are not able to respond in a timely manner, and (iv) are not flexible enough to deal with changing environments and situations with multiple autonomous actors. An alternative is to use a distributed planning approach in the form of a multi-agent system (MAS). A MAS consists of several independent and autonomous control

units (agents) that pursue their own interests and interact with other agents in the environment. As stated by MES et al. (2008), MASs are believed to be particularly suitable for decentralized systems in real-time and dynamic environments.

One of the key challenges is the configuration of the agents such that their self-interested behavior yields a near-optimal solution for the network as a whole. The network is defined as a closed transportation network consisting of a fixed number of pick-up and drop-off (P/D) locations. Yard tractors (YTs) transport the goods (e.g., semi-trailers) between these locations using a certain track layout. The network is closed, so no YT can enter or leave the system, even when idling. Similarly to EBBEN (2001), this Automated Transportation Network is defined as a fully automated system for the transportation, loading and unloading of goods using YTs supported by a MAS that functions as a planning and control system.

The goal of this paper is to design a flexible simulation model for fully automated trailer docking systems, using agent-based control, that is suitable to analyze different kinds of DCs (and terminals), varying in size, layout and/or modality. The loading/unloading operations at the system boundary will vary in practice. For example, picking up a semi-trailer and rearward docking is a complex task, which requires well-designed control systems depending on the semi-trailer characteristics. In the sequel, we use the term cargo to denote freight using any mode of transport that can be decoupled from the transporting entity. For example, semi-trailers and ISO-containers fall within this definition, but rigid trucks do not, as the cargo cannot be decoupled from the truck. Here, we specifically focus on the pick-up and docking of semi-trailers by means of YTs in a collision- and conflict free environment in such a way that it yields a cost-effective, near-optimal solution. The YTs within the system perform the following main actions: (i) pick-up arriving cargo that is decoupled from the truck (ii) move the cargo to a predetermined location, (iii) perform rearwards docking of the cargo into the loading dock, (iv) after the terminal has finished unloading/loading, pick-up the cargo, and (v) drop-off the cargo at a predetermined location such that a truck is able to pick it up again. We validate the system using discrete-event simulation, and using a case study at a Dutch logistics service provider.

The remainder of the paper is structured as follows. Section 2 presents our case study. In Section 3 we present our MAS. Next, in Section 4, we present the simulation model to analyze the proposed MAS and provide numerical results in Section 5. We close with conclusions in Section 6.

2 CASE DESCRIPTION

This study is motivated by the logistics service provider Royal Rotra based in Doesburg, The Netherlands. Rotra aims to build a new multimodal cross-dock in Velp, The Netherlands. The terrain will feature a $15,000\ m^2$ cross-dock with 150 loading docks as well as a container terminal. This location is a perfect pilot location to test the use of autonomous YTs and a MAS for planning and control because (i) there are no public roads (no regulatory restrictions), (ii) there is enough space (no capacity restrictions), (iii) there is a possibility to extend the system to multimodal (road and water) and (iv) there are no design restrictions (the terminal

has yet to be built). Including an industry partner as Rotra within this research enables us to incorporate many practical requirements at an early stage in the design to overcome possible design flaws at a later stage.

Using the case presented by Royal Rotra, we focus on a truck terminal (or DC) where semi-trailers arrive and depart by truck and are handled at the DC by YTs. The primary processes for the handling of semi-trailers at a DC are shown in Figure 1 (where 'trailer' is shorthand for 'semi-trailer').

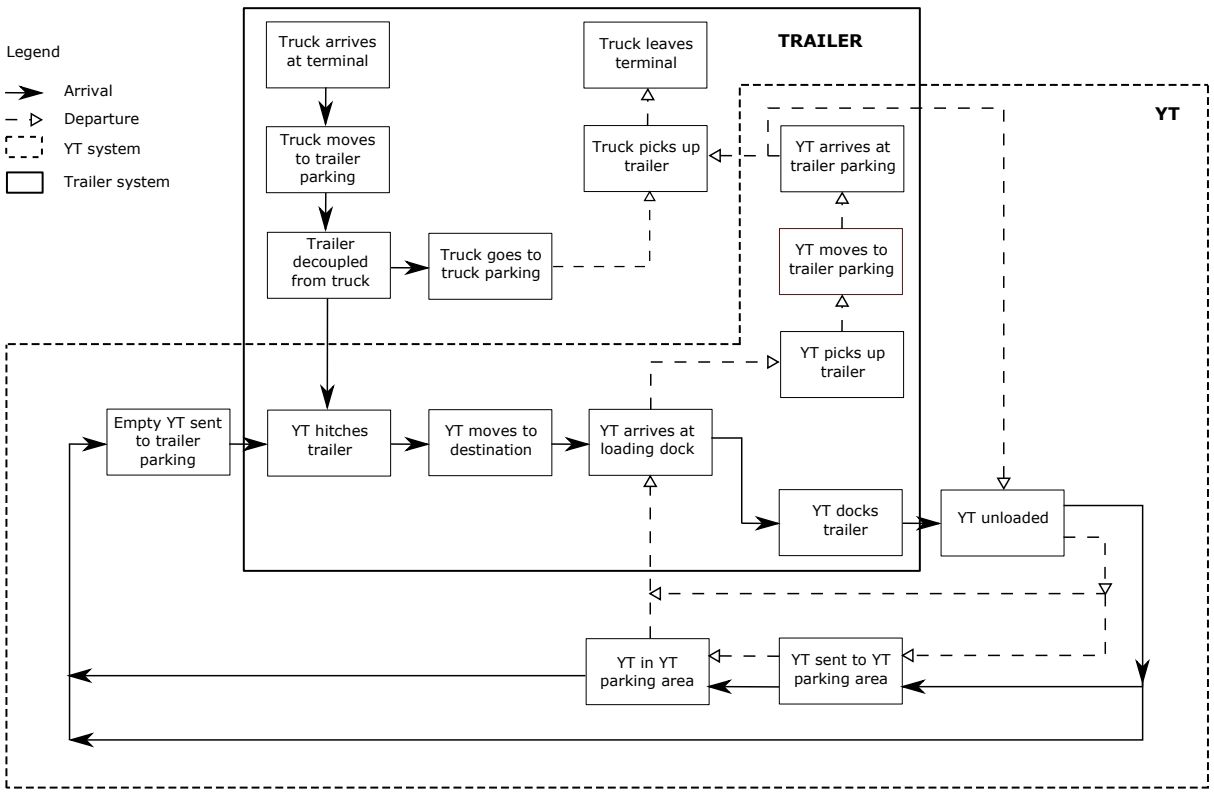

Figure 1: Primary processes of an Automated Transportation Network at a truck terminal.

3 SYSTEM DESIGN

Designing a MAS is a complicated and iterative process. Large-scale systems may contain agents in the hundreds operating at real-runtime, perceiving, communicating, negotiating, decision-making and executing. Such a complex system has to be designed properly to not only include all functionalities necessary to keep the system running and accurate, but also to make the system scalable and generic (i.e., applicable to a wide range of situations). In order to systematically address this challenge, a comprehensive and detailed methodology should be used to guide us through the process of breaking down the high-level objective of 'building an intelligent multi-agent system' into smaller, easier to grasp chunks, such that the system can be fully understood and then designed. The methodology should not only include high-level steps such as 'specify the system boundaries', but also mid- and lower level steps providing enough detailed guidelines to design and implement a software package based on intelligent agents.

For our case study, a MAS design methodology is needed that fully comprehends the de-

sign and analysis ranging from high- to low level steps. The comprehensive book of Wooldridge (2009) provides an overview of multiple methodologies, including AAII, Gaia, Tropos, Prometheus and Agent UML. We decided to use the Prometheus methodology, since it is generally conceived as a comprehensive, practical and easily implementable methodology specifically for designing multi-agent systems, providing everything necessary for specifying and designing the agents (Winikoff and Padgham, 2004). Throughout the literature, Prometheus is used to design multi-agent systems, including systems that feature AGVs (Erol et al., 2012; Wu et al., 2012). An important practical aspect is that the Prometheus Design Tool (PDT) is freely available. Moreover, it has built-in completeness and consistency checks. Prometheus is based on industry standards like case scenarios, UML sequence diagrams, AUML (itself an extension of UML) and the Rational Unified Process (RUP) (Padgham and Winikoff, 2004).

The Prometheus Method consists of the following three phases: (i) the system specification phase that focuses on identifying the goals and basic functionalities of the system, along with inputs (percepts) and outputs (actions), (ii) the architectural design phase that uses the outputs from the previous phase to determine which agent types the system will contain and how they will interact, and (iii) the detailed design phase that looks at the internals of each agent and how it will accomplish its task within the overall system (Padgham and Winikoff, 2004). In the following sections, we summarize the results of the different design phases.

3.1 System specification

During the system specification phase, we breakdown the overall goals of the system into smaller sub goals to elucidate which kind of behavior and functionality is required to reach the goals of the system as a whole. From this breakdown, we define the following (generic) grouping of functionalities:

1. Demand Management (DM) - This functionality monitors inbound/outbound cargo, obtaining information about expected arrival/dispatch time and cargo description.

2. Park Management (PM) - This functionality assigns pick-up and drop-off locations to all cargo and YTs.

3. Vehicle Scheduling (VS) - This functionality decides when and where which YT should pick-up and drop-off cargo or empty semi-trailers.

4. Vehicle Routing (VR) - This functionality determines the route such that the YT can pick-up and drop-off cargo or empty semi-trailers.

5. Conflict Resolution (CR) - This functionality monitors all YT movements and makes sure there are no collisions and resolves conflicts.

6. Battery Management (BM) - This functionality determines when and where YTs should be refueled or recharged.

The system specification phase continues with scenario development. Every scenario sets out a state the system can be in and translates this into how the functionalities defined above should

act upon this state. We define the following scenarios: (i) cargo arrival, (ii) cargo departure, (iii) empty cargo movement, (iv) YT idle, (v) YT low battery, and (vi) YT conflict. The first scenario assumes inbound cargo and the second outbound cargo. The third scenario occurs when empty cargo is moved to (or removed from) a dock. The fourth scenario occurs when a YT is idling and thus has to be parked somewhere. The fifth scenario occurs when a YT has a low battery and thus has to be recharged at the appropriate location. Note that recharging can also be replaced by refueling (in case of a combustion engine). The final scenario is one that can happen anytime during system run-time and thus also during other scenarios, namely YTs that are in conflict. This could for example happen when two or more YTs want to take the same route at the same time. Appropriate measures have to be taken when this occurs such that congestion and system dead- or livelocks are avoided.

3.2 Architectural design

The architectural design phase defines the agents within the system based on the functionalities and scenarios defined in the previous design phase. In this phase we determine how agents interact with each other and which external connections are required. We employ a data coupling diagram to provide insight into the connections between functionalities based on the scenarios defined. Directed links are shown between functionalities and data. Arrows pointing towards data indicate that data is written by the functionality and the other way around indicate the usage of data. Double-headed arrows indicate the usage and writing of data by the functionality. Bold arrows indicate important connections within the system. All data used and produced by the system must be in the diagram somewhere including all external data sources. This also functions as a consistency check between the first two design phases. For brevity of writing we do not show the internal capabilities of the functionalities in the data coupling diagram (see Figure 2).

The main external database required for the MAS is the Transport Management System (TMS). This TMS contains information about inbound and outbound cargo, cargo properties (e.g., consolidation information), where the cargo should go to at the DC (originating from the planning system) as well as client-specific information. The expected arrival time of inbound cargo is obtained using the board computer of the transporting entity. From Figure 2 we see seven agents emerging. We further elaborate on these agents in Section 3.5.

3.3 Detailed design

The final design phase builds upon the agent descriptors defined in the previous phase and further specifies the behavior of every agent. This includes all triggers the agent responds to, how it responds accordingly, and which agents it interacts with. This phase narrowly defines the external connections of the MAS as these contain many of the triggers the MAS responds to (e.g., 'new arriving transport' or 'semi-trailer unloaded'). An important part of this final phase is the outline of what kind of intelligence an agent should inhabit. For example, the Vehicle Scheduling Agent should contain a scheduling algorithm or a set of rules which makes sure the YTs are scheduled in an efficient and effective way. To show how the MAS fits within an automated YT system, we employ the framework of Le-Anh and De Koster (2006). This gives

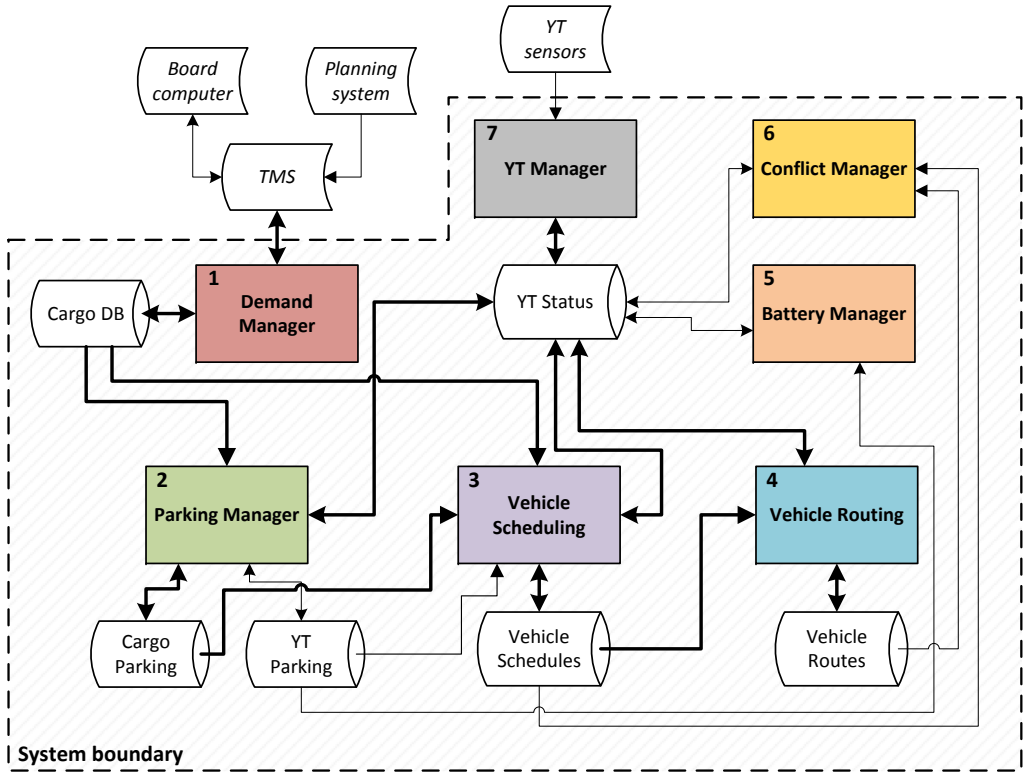

Figure 2: Data coupling diagram of the MAS.

us a solid theoretical framework on how to align the design of the MAS (that corresponds with the tactical and operational levels in the framework) with the design of a physical automated YT system. We first focus on the strategic level in Section 3.4 and connect the agents with the tactical and operational levels in Section 3.5.

3.4 Guide-path design

One of the key design issues is the design of the guide-path (or flow path). The guide-path is a layout of trajectories that YTs can follow by using, for example, wires in the ground connecting all P/D locations (i.e., loading docks and parking slots). The design of the guide-path lay-out can be done in various ways (Vis, 2006) and has a direct impact on system performance (such as travel time, number of required vehicles and degree of congestion). Specifically, it provides bounds to where YTs are able to drive. As the lay-out of the building and the locations of the P/D points are relatively straightforward, we fix these as input factors. The cross-dock in our case study will have a rectangular shape with 75 loading docks (P/D locations) at opposite sides. In this paper, we focus on one side of the cross-dock only. The key design issue is to create a well-balanced trade-off between the space required for the lay-out and the most efficient way of maneuvering, such that it can accommodate a multitude of vehicle configurations.

We use a unidirectional single loop to connect the main areas on the terminal and extend it with bidirectional arcs per P/D location to (i) remove the YT and cargo from the single loop to lower congestion and (ii) form a guide-path for the rearward docking/parking maneuver. Because we deploy articulated vehicles (a combination of a YT with a semi-trailer) we need maneuvering space to straighten out the semi-trailer behind the YT when turning. From here

on we call this maneuvering space before a dock or parking slot a crossroad. The crossroads facilitate rearward docking and parking maneuvers and are thus by definition bidirectional arcs. This makes the entire system a mixed uni-bidirectional guide-path. Figure 3 exemplifies how the crossroads fit within the guide-path design and shows the overall design of the stackyard at our pilot location.

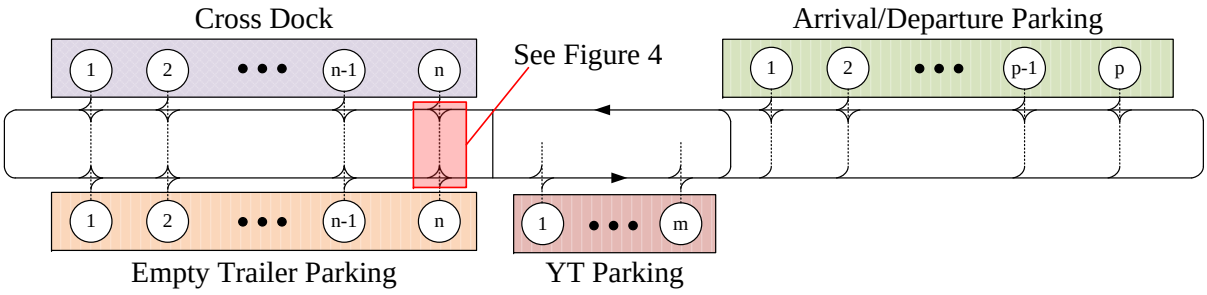

Figure 3: Guide-path design and layout of the pilot location.

At the Arrival/Departure Parking, truck drivers drop-off and pick-up semi-trailers where p parking slots are available. From this point on the autonomous YTs take over the handling at the terminal and use the guide-paths to dock, park, and move around semi-trailers. The cross-dock has n docks and the parking opposite to it also has n parking slots. The YT parking is used for charging YTs and parking when YTs are idling, and is strategically positioned between the main areas to decrease the response time of a YT. The YT parking has m parking/charging slots.

There are multiple ways of designing the crossroads of which the three most interesting are further discussed here. We limit our discussion of the crossroads to the area in front of the cross-dock with a parking area opposite to it (e.g., to temporarily park semi-trailers). Figure 4 shows three different crossroad designs.

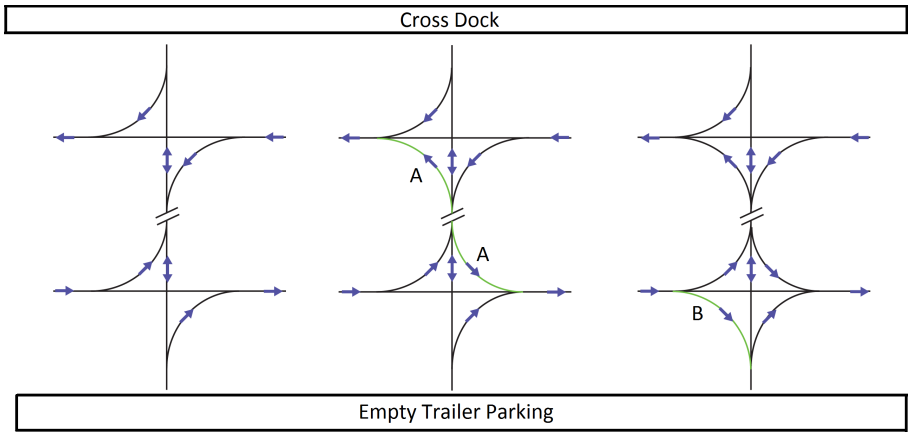

Figure 4: Three different crossroad designs. Left: design 1; middle: design 2; right: design 3.

To illustrate, a docking maneuver starts with a YT carrying a semi-trailer at the top main road on the right-hand side of the dock. The YT turns away from the main road facing south until the back of the semi-trailer is facing the dock. The YT then backs up the semi-trailer until it is docked into the (un)loading area. When the YT is decoupled it can continue its path

using the curve facing away from the cross-dock in western direction. When picking up a docked trailer, a similar maneuver is used.

The design on the left in Figure 4 (design 1) is the most simple design where only a crossroad is incorporated and there is no possibility to move between the top and the bottom road (or vice versa). This limits the ability to take shortcuts and thus the entire length of the cross-dock has to be traversed before the YT can switch between the main roads. Design 2 resolves this problem by added two additional arcs (marked with an A) such that YTs can move between the two roads at every P/D location. Design 3 further extends the ability to take shortcuts by adding an arc at every parking slot (marked with a B). The added value of more complex designs depends on many factors, such as the utilization of the docks and parking areas and preference for forward driving. The different designs can be analyzed using the proposed simulation model (Section 4). For this case study, we choose for design 2.

3.5 Agent intelligence

Using the framework of Le-Anh and De Koster (2006), we finalize the MAS design by designing the intelligence of the agents such that they fulfill their design objectives. Since the focus of this paper is on the evaluation framework and not on the detailed design issues, we only briefly describe this design for the seven agent types.

- Demand Manager - This agent does not require complex algorithms, but instead provides the link between external systems (e.g., TMS) and the MAS. It is however of utmost importance that all data required is readily available and up-to-date.

- Parking Manager - We use a nearest-available rule for all parking areas. Arriving and departing semi-trailers are assigned to the first available (i.e., not occupied) parking slot.

- Vehicle Scheduling Agent - We need to assign YTs to trailer movements. A common way of supporting assignment decisions in MASs is through the use of an auction mechanism where YTs compete for orders. The Vehicle Scheduling agent initiates a proposal when a new job enters the system via the Demand Manager (e.g., pick-up at cross-dock). All YT agents evaluate this proposal and send back a bid. Based on a bid evaluation function the winner is announced. We use the auction mechanism as described in MES et al. (2007).

- Vehicle Routing Agent - The routing agent determines the shortest route the YT should take given the pick-up and drop-off location of the job.

- Battery Manager - The battery management agent uses an opportunity charging strategy as defined by McHaney (1995). Whenever a YT is idling it is sent to the YT parking. The Battery Manager also monitors the battery level of all YTs and asks the Vehicle Scheduling agent to schedule a charging job whenever the battery is below a certain threshold.

- Conflict Manager - This agent is responsible to avoid collisions, congestion and deadlocks. We use a priority list to make stop-and-go decisions based on the current status of the YT. When two or more YTs want to use the same arc at the same time, the Conflict Manager

evaluates which YT has priority based on the current jobs the YTs are processing and stops the YT with the lowest priority.

- YT Manager - The YT Manager agent has an important role within the MAS as it feeds information about the YT to the system (e.g., current position and battery level). It furthermore uses the bid calculating function as described above to respond to the proposals of the Vehicle Scheduling agent.

4 SIMULATION MODELING

To evaluate the proposed MAS, we use discrete-event simulation. We present a flexible simulation model following the same logic as the agent-based design presented in Section 3. Although we focus on our case study, we provide enough flexibility in our model to tailor the simulation model to other DCs. Two important parts of our simulation model are (i) the modeling of a track system that resembles the guide-path design and (ii) incorporating the agent intelligence. In the following subsections we focus on these two parts.

4.1 Modeling the guide-paths

We model the guide-paths using so called *Tracks*. These tracks allow moving units (i.e., YTs) to travel from one node to another and are characterized by length, direction, and curvature. The complete guide-path is composed by a large set of connected tracks. The crossroads are modeled as bidirectional arcs and all other tracks as unidirectional. One of the challenges is building a flexible track system. Since we want to evaluate various physical designs, inserting the entire track system manually would not be an option. Therefore, we created a procedure (*CreateTracks*) that generates the whole guide-path layout using the given input. For our case study, this given input consists of: the values of n, m, and p from Figure 3, the distance between the docks, the distance between the parking slots, the length of the crossroads, the length of the parking slots, and the required distance between the cross-dock and the main road.

The network creation procedure enables us to systematically assess various designs in order to come up with an appropriate design for our case study. Furthermore, this way our model is more flexible to be used at other DCs. An example of the result of *CreateTracks* using five docks is shown in Figure 5. The tracks on the right side connect the cross-dock with the Arrival/Departure Parking and the YT parking, see Figure 3.

4.2 Modeling the MAS

A crucial part of the simulation model is the agent-based planning and control mechanism. We program the functionalities of every agent as described in Section 3.5. Communication between agents lies in the nature of a MAS and is also incorporated into our model. In a sense, the simulation model not only serves as a tool to analyze the physical design and MAS control, but also to validate the MAS software architecture. An overview of the simulation model, including the initialization and the communication between agents, is shown in Figure 6. In this figure the green rounded shapes are triggering events and the red rounded shapes are end events. The

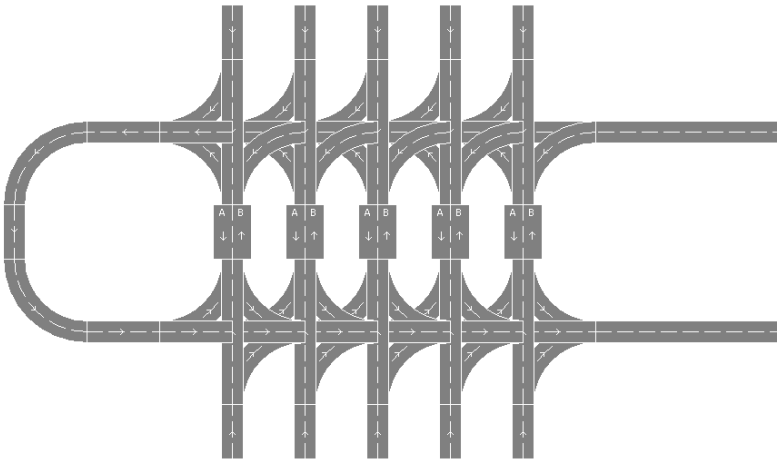

Figure 5: Result of the *CreateTracks* procedure.

processes are depicted using rectangles. We use dotted lines to indicate when an agent has to wait for a triggering event that has not been included to ease the presentation.

5 IMPLEMENTATION & RESULTS

We implemented our model in the discrete-event simulation software package Tecnomatix Plant Simulation. The basis is the layout of the pilot location in our case study enabled by Royal Rotra. The semi-trailers are shown as small images of the semi-trailers currently used by Rotra. We distinguish between unloaded and loaded transport with a small pallet icon in the top right corner of every semi-trailer. The YTs move over the guide-path and can hitch and unhitch semi-trailers at the appropriate locations. The docks are modeled as transfer stations, which function either as an unloading station or as a loading station, based on the job at hand. The inner area of the cross-dock is modeled as buffers per dock. These buffers contain the loads that have been unloaded. Figure 7 shows a screenshot of the simulation model.

We omitted the YT parking from the screenshot to get a clearer view of the model. In this figure we see three deployed YTs. The YT on the left in the top frame is currently hitching a departing semi-trailer whereas the YT on the right in the same frame is currently transporting a departing semi-trailer. In the bottom frame, we see a YT driving rearwards to pick up an arriving semi-trailer and a truck on its way to drop-off a semi-trailer at the Arrival/Departure Parking. The remainder of this section describes the verification and validation of the simulation model and some preliminary results.

Before the simulation model can be used, it has to be thoroughly verified and validated. Verification of our model was done using (i) code debugging, (ii) model reviewing, (iii) providing demonstrations to the stakeholders, (iv) comparing the assignment of the YT to a semi-trailer with the actual YT picking it up, and (v) watching the animation of the semi-trailers moving along the guide-paths. From this analysis, we conclude that the simulation is an accurate translation of the conceptual model.

We validate our model using the techniques described in Robinson (2014). We illustrate the validation process using the Parking Manager. As described before, the Parking Manager

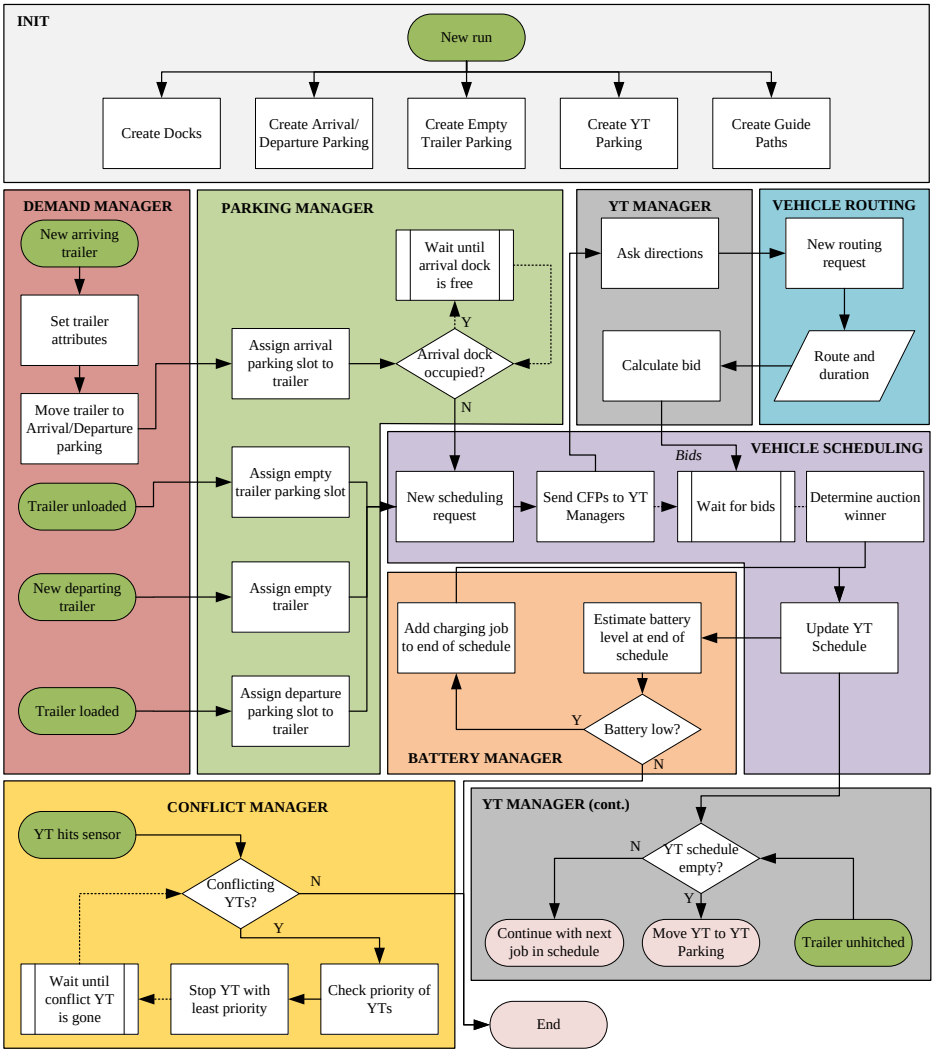

Figure 6: High level overview of the simulation model.

uses a nearest available rule to assign parking slots to semi-trailers. We validated this agent by running the model and checking whether the parking slot assignment corresponds with the nearest available rule. Figure 8 shows the assignment of parking slots to (i) arriving semi-trailers at the Arrival/Departure parking and (ii) unloaded semi-trailers at the Empty Trailer (ET) parking. We expect from the Arrival/Departure parking that initially, arriving semi-trailers have a low parking slot ID because the time it takes for the YT to pick-up the semi-trailer is lower than the inter-arrival time of the arriving semi-trailers. At some point the system is full and semi-trailers are also starting to depart. This results in a mixture of arriving and departing semi-trailers at the Arrival/Departure Parking. We thus expect that the parking slot IDs will then be higher, also because there are more jobs at hand for the YTs and thus the time it takes for the YT to pick-up or drop-off a semi-trailer increases. From Figure 8 we conclude that this is indeed the case. For the ET parking, the nearest available slot is opposite to the dock where the semi-trailer has been unloaded. We thus expect that, under normal conditions, the parking slots are assigned opposite to the arrival dock. However, in case of increasing utilization of the ET parking, only a few parking slots are available and thus the nearest available parking slot

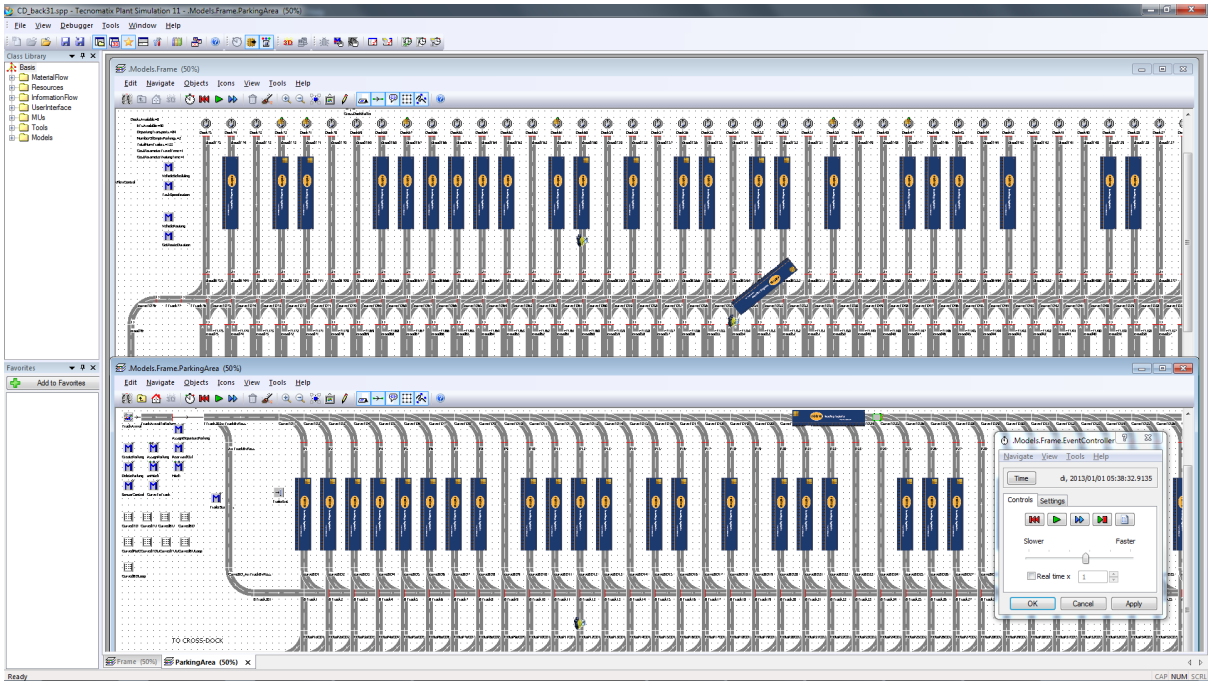

Figure 7: Screenshot of the simulation model.

may be far away from the arrival dock. When the ET parking is full, the model waits until a parking slot becomes available and assigns this slot to the semi-trailer that has been waiting for the longest time, irrespective of its location. Hence, in case of high utilization, we expect major outliers. From Figure 8 we conclude that our expectations of the ET parking correspond with the results. We used similar procedures to validate the other agents.

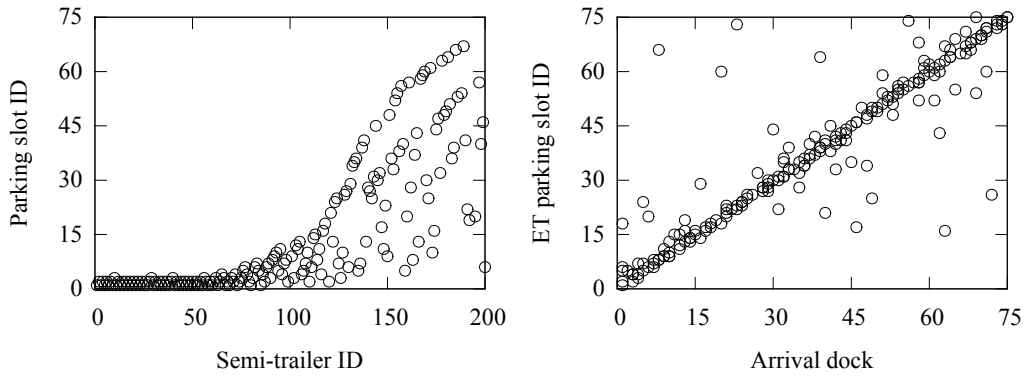

Figure 8: Scatter plots of the assignment of parking slots to semi-trailers.

To illustrate the working of our simulation model, we perform a few experiments considering varying numbers of YTs. For every run, we did ten replications each with 200 arriving and departing transports. The results are summarized in Table 1. The cycle time, shown in hours, consists of waiting time, (de)coupling time, and (un)loading time. The travel time is omitted as this is just a minor component of the cycle time. We see that the waiting time significantly decreases when 3 instead of 2 YTs are deployed. The waiting time at the ET parking shows less volatility as this also includes the time the semi-trailers wait for a loading job. We conclude from this preliminary analysis that a minimum of 3 YTs are required to reach an acceptable

cycle time.

Table 1: Simulation results.

	3 YTs					2 YTs				
	Mean	SD	Min	Max	95%	Mean	SD	Min	Max	95%
Total Cycle time	7:35	2:39	1:50	16:40	11:53	12:51	4:55	3:39	23:59	20:13
Waiting for YT at A/D	1:32	2:13	0:00	7:56	6:05	4:46	5:11	0:00	19:16	15:04
Waiting for YT at CD (arrival)	0:27	0:27	0:00	2:25	1:21	0:56	0:46	0:00	4:03	2:38
Waiting at ET	2:58	2:22	0:03	14:34	7:47	4:11	3:53	0:01	22:01	12:23
Waiting for YT at CD (departure)	0:40	0:32	0:00	2:29	1:44	1:03	1:00	0:00	4:01	3:11

6 CONCLUSIONS

We presented a simulation model of a generic automated planning and control system based on agent technology for the pick-up and docking of semi-trailers by means of Yard Tractors (YTs) in a collision- and conflict free environment. We designed a Multi-Agent System (MAS) suited for the automatic handling of semi-trailers using YTs, which we have put to work using our pilot location at a Dutch logistics service provider. Next, we specified the design of the simulation model. We concluded with the implementation and validation of the simulation model and some results.

ACKNOWLEDGMENTS

This research is supported by the Dutch organization for scientific research (NWO) through the RAAK-PRO project: “Intelligent Truck Applications in Logistics” (INTRALOG).

References

Ebben, M. J. R. (2001). *Logistic control in automated transportation networks*, PhD thesis, University of Twente, Enschede, The Netherlands.

Erol, R., Sahin, C., Baykasoglu, A. and Kaplanoglu, V. (2012). A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems, *Appl. Soft Comput.* **12**(6): 1720–1732.

Le-Anh, T. and Koster, R. de (2006). A review of design and control of automated guided vehicle systems, *European Journal of Operational Research* **171**(1): 1–23.

McHaney, R. (1995). Modelling battery constraints in discrete event automated guided vehicle simulations, *International Journal of Production Research* **33**(11): 3023–3040.

Mes, M., Heijden, M. van der and Harten, A. van (2007). Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems, *European journal of operational research* **181**(1): 59–75.

Mes, M., Heijden, M. van der and Hillegersberg, J. van (2008). Design choices for agent-based control of agvs in the dough making process, *Decision Support Systems* **44**(4): 983–999.

Padgham, L. and Winikoff, M. (2004). *Developing Intelligent Agent Systems: A Practical Guide*, John Wiley & Sons, Inc., New York, NY, USA.

Robinson, S. (2014). *The Practice of Model Development and Use*, 2 edn, Palgrave Macmillan.

Vis, I. F. (2006). Survey of research in the design and control of automated guided vehicle systems, *European Journal of Operational Research* **170**(3): 677–709.

Winikoff, M. and Padgham, L. (2004). The prometheus methodology, *in* Bergenti, F., Gleizes, M.-P. and Zambonelli, F. (eds), *Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook*, Springer US, Boston, MA, pp. 217–234.

Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*, 2nd edn, Wiley Publishing.

Wu, X., Lou, P., Cai, Q., Zhou, C., Shen k. and Jin c. (2012). Design and control of material transport system for automated guided vehicle, *International Conference on Control 2012*, UKACC, pp. 765–770.

AUTHOR BIOGRAPHIES

BERRY GERRITS is a prospective PhD candidate within the department of Industrial Engineering and Business Information Systems at the University of Twente, The Netherlands. He received a MSc in Industrial Engineering in 2016. His research interests are transportation logistics, multi-agent systems, supply chain management and business process optimization. His email address is b.gerrits-1@student.utwente.nl.

MARTIJN R.K. MES is Associate Professor within the department of Industrial Engineering and Business Information Systems at the University of Twente, The Netherlands. He holds a MSc in Applied Mathematics (2002) and a PhD in Industrial Engineering and Management at the University of Twente (2008). His research interests are transportation, multi-agent systems, stochastic optimization, discrete event simulation, and simulation optimization. His email address is m.r.k.mes@utwente.nl.

PETER C. SCHUUR is Associate Professor within the department of Industrial Engineering and Business Information Systems at the University of Twente, The Netherlands. He received a PhD in Mathematical Physics from the University of Utrecht, The Netherlands. On the theoretical side his research interests are on: packing, covering and cutting problems, game theory, and simulated annealing. On the applied side his research interests are on (closed loop) supply chain management, vehicle routing, distributed planning, multi-modal networks, dynamic pricing, reverse logistics, layout problems, and warehouse modeling. His email address is p.c.schuur@utwente.nl.