ERASMUS UNIVERSITEIT ROTTERDAM

Erasmus School of Economics

Operations Research and Quantitative Logistics

Master Thesis

# Autonomous Driving on Container Terminals

Author: Pim van Leeuwen (412793)

Supervisor: D. Galindo Pecin
Second Assessor: T.A.B. Dollevoet

September 17, 2019

**Abstract**

Terminals play an important role in the process of transporting containers from origins to destinations all over the world. Recent developments have led many terminals to explore the opportunities of automation, specifically those offered by Automated Guided Vehicles (AGV's). So far, AGV systems have primarily used central routing and driving methods to control movements of these vehicles. The aim of the system's strategy is to prevent both collisions and deadlocks and to maximize container handling speed and efficiency. This thesis investigates the possibilities of autonomous AGV's, which are responsible for making their own driving decisions and for coordinating with other vehicles around to prevent collisions and deadlocks. A decentralised strategy is developed and tested extensively by simulation experiments, the results of which show a performance that is roughly equal to that of a centralised strategy adopted by TBA's TIMESquare software.

# Preface

This thesis was written during a six-month internship at TBA in Delft. I am very grateful to have had the opportunity to work around inspiring professionals from whom I have learned so much within half a year of time. In April, my knowledge of operations at container terminals was negligible. Half a year later, I now have a good understanding of many of the processes, equipment types and strategies. Special thanks goes out to Arjen de Waal, my supervisor at TBA. You were always willing to personally answer my questions, which was very helpful.
Handing in this thesis also marks the end of my programme at the Erasmus University of Rotterdam. I would like to thank all professors and supporting staff for offering both fun and challenging courses over the past five year. Regarding my thesis, thank you Diego Galindo Pecin for supervising me along the way. Our sessions have really helped me gain insight in the mathematical and operational sides of the problem.
But most of all, I want to thank my fellow econometricians and friends: Arthur, Jeroen, Peter, reinieR and Tobias. Not only have we laughed our butts off the past five years, you guys have also had a bigger impact on my academic results than any professor. Finally, thanks to Bart Melman and Gert Kerkdeur for their feedback on an earlier version of this thesis.

*Pim van Leeuwen*

# Contents

# 1 Introduction

Transporting containers from origin to destination often happens in multiple steps. For long distances, containers are typically transported by barge, train or truck towards a deep sea vessel, which is responsible for most part of the journey. The last miles of transport are once again driven by train or truck. Terminals play an important role in the full process from origin to destination. Here, containers are taken from a barge, train or truck to be loaded onto a vessel, and vice versa. Between loading and unloading, a container may be stored in an area called the yard of the terminal.

Container handling on a terminal is split into a number of activities. Containers need to be loaded and unloaded to/from a vessel, truck or train, they need to be stacked in the yard, and they need to be transported to and from the yard. All these activities can be carried out by different types of handling equipment. Loading and unloading of deep sea vessels is done by large quay cranes (QC), which are considered a terminal's key asset. Transport to and from the yard happens with vehicles such as straddle carriers, shuttle carriers, reach stackers and terminal trucks. Some of these vehicles can stack containers in the yard, but this can also be done by specialised stacking cranes such as Rubber Tired Gantries (RTG). All these vehicle and equipment types typically have one thing in common, they are manually operated.

A development that container terminals have had to deal with over recent years is the growing size and capacity of vessels. New deep sea vessels may well carry over 20,000 twenty foot equivalent units (TEU). In order to load and unload such huge vessels quickly, container handling across the terminal has to happen in an efficient manner. Starting with the Europe Container Terminal (ECT) of Rotterdam in 1992, many terminals have embraced the opportunities that automation offers in order to reduce costs and increase efficiency.

For transshipment in particular, a commonly used piece of equipment is the Automated Guided Vehicle (AGV), which is an unmanned vehicle that follows a grounded guide path in order to travel to its destination. An AGV is coupled to other pieces of equipment, in the sense that it cannot load and unload itself like a reach stacker or straddle carrier. As a consequence, the AGV has to wait for a quay or stacking crane to (un)load it. Recently, decoupled AGV's have been developed that can take containers from racks, so that they do not have to interact directly with cranes. These are referred to as lift-AGV's.

Although each AGV is guided from origin to destination, there still needs to be coordination between AGV's in order to prevent collisions. Coordination of a fleet is usually centralised: a single, central operator is responsible for dispatching, routing and motion control. This thesis will consider the driving (motion) problem of a fleet of AGV's on a container terminal. Rather than a centralised approach, driving decisions are delegated to the vehicles themselves. Through on-board sensors and communication technology, collisions and conflicts have to be avoided and resolved. AGV routing is still done by the central computer, so the focus of this thesis is on autonomous driving. The main research question is: to what extent would efficiency of container handling on a terminal change when switching from centralised AGV drive control to an autonomous strategy?

Advantages of decentralised driving include increased robustness to failures and dropouts. This thesis aims to find an autonomous driving strategy that can compete with a chosen centralised strategy in terms of operational performance and efficiency. Through simulation experiments, the decentralised strategy proposed is compared to TBA's state of the art centralised strategy in their TIMESquare software. Simulation results show that the proposed autonomous strategy yields a quay crane productivity equal to that of the centralised strategy.
The academic contributions of this thesis include:

- an analytic speed optimisation for trajectories with known distance and desired beginning and ending speed
- a novel decentralised collision avoidance approach that also prevents deadlocks
- simulation experiments to compare two driving strategies

The remainder of this thesis is organised as follows. Section 2 provides an overview of previous research done in the area of AGV systems. Section 3 describes the problem considered and gives a mathematical formulation. Section 4 presents the methods used for request assignment and routing. The scope of this thesis is mostly limited to driving, which is discussed extensively in Section 5. Section 6 discusses the simulation setup and the scenario used for testing, after which results are presented and discussed. Also in this section, sensitivity analysis is performed. Finally Section 7 concludes, discusses limitations of the methods used and suggests a number of possible extensions.

## 2 Literature review

AGV systems have been studied in the scientific literature ever since practical applications started. The process that starts with receiving a transshipment request and ends with an AGV arriving at the given destination can be divided into three steps of optimisation:

1. Dispatching: assigning the request to an AGV
2. Routing: determining the path that the assigned AGV follows from origin to destination
3. Driving: controlling the speed of an AGV along its route in order to prevent collisions with other vehicles

To find an optimal solution, these three problem should be solved jointly. However, due to computation time considerations, this is typically not feasible and dispatching in particular is done separately, prior to solving the routing and driving problem. Multiple dispatching strategies are studied and tested by simulation in (Grunow, Günther, & Lehmann, 2006). A realistic experimental scenario is used that reflects real-life conditions on an automated container terminal. The best strategy found has an optimality gap of less than 5 percent.

For this thesis, however, request assignment is not given as much attention as routing and driving. Central routing algorithms have been developed that plan a route for each request by an AGV. In (Liu, Jula, Vukadinovic, & Ioannou, 2004), a complete centralised AGV system is studied, including a method for dispatching, routing and vehicle interaction rules. Experiments are performed for two characteristic, real-life container terminal layouts. The remainder of this section discusses a selection of articles in which elements of AGV routing or driving are considered in either a centralised or distributed setting.

### 2.1 Routing review

A problem frequently encountered in AGV systems is that routing AGV's by finding shortest distance paths results in deadlocks. If congestion on paths is ignored when routing, AGV's will have to wait for each other in order to avoid collisions. This creates the possibility of a circular situation in which AGV's have to wait for each other in such a way that no vehicle can move: a deadlock. One way to detect and avoid deadlocks is by modelling the AGV system as a Petri net (Wu & Zhou, 2001). A Petri net is a special type of directed graph that can be denoted by a quadruple $PN = (P, T, I, O)$. Here, $P$ is a (finite) set of places and $T$ is a set of transitions such that $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$. $I : P \times T \rightarrow \{0, 1\}$ and $O : P \times T \rightarrow \{0, 1\}$ are respectively input and output functions. AGV's are modelled as tokens that can travel between places through transitions. Each time an AGV tries to use a scarce resource (i.e. an arc or edge in the graph), the central controller checks whether this transitions creates a deadlock. Although this approach has been studied extensively in the literature, Petri nets are more suitable for static environments. Using them in a dynamic resource allocation problem for deadlock prevention may lead to unacceptable computations times (Viswanadham, Narahari, & Johnson, 1990).

An algorithm for deadlock avoidance using a bipartite resource-allocation graph $G = (V, E)$ is presented in (Yoo, Sim, Cao, & Park, 2005). The vertices in $V$ represent the AGV's on one side, and zones in the AGV path layout on the other. The paths of $G$ are given in a $|V| \times |V|$ matrix $P$, such that each element $(i, j)$ of $P$ represents the number of paths from $i$ to $j$, with $i, j \in V$. With basic linear operations, the authors develop a deadlock avoidance algorithm that is more robust than Petri net methods and that requires less computation time.

Besides deadlocks, an AGV control system should also avoid collisions. In general, this can by done by i) zone control, ii) forward sensing AGV's or iii) a priori collision avoidance in route planning. Zone control is a strategy that divides the system environment into multiple non-overlapping zones (Ho & Liao, 2009). The presence per zone is restricted to a maximum of one AGV at a time. When equipped with front-end sensors, AGV's can sense (moving) objects blocking their path and respond fittingly, possibly assisted by a central controller (Le-Anh & De Koster, 2006).

For developing an autonomous AGV system in which vehicle control is decentralised, forward sensors on AGV's are an adequate way to avoid head-on and head-to-tail collision. However, for other types of collisions, side sensors may be required, as well as some sort of communication between neighbouring vehicles. In addition, it is interesting to look at methods to avoid collisions a much as possible in a priori route determination. In (Downsland & Greaves, 1994), a corrective approach is used. First, an AGV is routed to its destination using the shortest distance path from its origin, after which this path is checked for collision. If a collision is detected, there are three options: introducing a delay during the route, making a small detour, or finding a completely new path. Although this method is fast for small networks with few vehicles, it has not been shown to also work well on a large area such as a container terminal, with many AGV's.

Finding routes for all vehicles such that joint travel time is minimised forms a complicated combinatorial problem. In fact, it can be shown that this multi-agent route planning problem is NP-hard, meaning optimal routes cannot by found in polynomial time, unless P = NP (Ter Mors, 2009). Consequently, for large instances, computation time will not be acceptable for real-life applications. It is possible, however, to find the optimal route for a single AGV, given the routes of all other vehicles, in polynomial time. For a dynamic routing problem, AGV routes can be planned sequentially (Möhring, Köhler, Gawrilow, & Stenzel, 2005).

Let $G = (V, A)$ be the terminal representing graph and let $t_a$ be the (deterministic) traversing time on arc $a \in A$. Let $S(a) \subset A$ be the set of arcs that are blocked when the center of an AGV is on arc $a$. If (the center of) an AGV traverses the arc $a$ during the interval $[t_1, t_2]$, no other AGV may be present on any arc $u \in S(a)$. In this way, AGV route planning sequentially creates time windows on the arcs of $G$. More importantly, the dimensions of AGV's can be disregarded, since these are represented in the sets $S(a)$. As a result, satisfying a request $p \in P$ breaks down to routing a infinitesimal point, representing the center of the appointed AGV, from $o_p$ to $d_p$. As an additional constraint, a set of time windows on each arc needs to be respected. This problem is known as the Shortest Path Problem with Time-Windows (SPPTW) (Möhring et al., 2005).

To solve the SPPTW, the authors develop an arc-based Dijkstra algorithm that finds an optimal route in polynomial time, given all previously planned routes, which are treated as fixed. However, the authors assume a constant speed for all vehicles. This is an unrealistic assumption, since AGV's need to accelerate from their starting position and decelerate to standstill when nearing their destination. Also, the curve speed of an AGV is typically lower than its speed when travelling in a straight line. Realistically, the traversing time $t_a$ of arc $a$ should depend on the speed of the traversing vehicle on its previous arc.

## 2.2 Driving review

Even when a vehicle knows exactly which path to follow to its destination, many decisions still have to be made in order to actually arrive there safely. The speed of an AGV along its route, as well as its interactions with other vehicles, needs to be controlled. In centralised systems, a central agent controls the speed of each AGV and also decides which vehicle has priority in case there is a conflict on an intersection. A zone-controlled, real-time, centralised system is proposed to avoid both collisions and deadlocks (Fanti, 2002).

In decentralised systems, AGV's are responsible for controlling their own speed in order to avoid collisions. Conflicts on intersections are resolved by predetermined priority rules, or by communication between AGV's in the form of negotiating or claiming, although these two methods are not mutually exclusive. Advantages of decentralised approaches are (de Campos, Falcone, & Sjöberg, 2013):

1. AGV's can coordinate in order to trade-off individual and global objectives, while avoiding conflicts.
2. Robustness to failures of vehicles or communication can be guaranteed.
3. Dynamic communication features such as low data rates, drop-outs or proximity bases communication can be handled adequately.

Treating routes as given, (de Campos et al., 2013) define states for agents that represent their position and velocity. Modelling intersections as a critical set of overlapping positions across agents, the authors argue that collisions are prevented as long as only one agent may be present in the critical set at the same time. In other words, the intersection may be traversed by one vehicle at a time. As a control policy, a convex optimisation problem is solved sequentially for all agents. The authors report several advantages such as short computation time, low complexity and scalability, but the results only consider feasibility conditions. No optimality or efficiency arguments are made.

An approach to improve intersection efficiency of autonomous passenger cars is given in (Dresner & Stone, 2008). In this model, the agents drive autonomously, but traffic at intersections is controlled by an intersection manager. Using simulation, the authors demonstrate that their adapted First Come First Served policy significantly outperforms current intersection control strategies, which uses traffic lights and stop signs. This approach is very useful for urban traffic management, where intersections are easily identified and separated. On container terminal, however, intersections may effectively happen anywhere across the horizontal roads. Another solution method is therefore required.

In (Manca, Fagiolini, & Pallottino, 2011), a decentralised, multi-vehicle system is proposed in which autonomous vehicles compete for scarce resources. A predetermined path for a vehicle consists of a sequence of nodes that are travelled in order. Individual nodes constitute the micro level resources, whereas sequences of nodes (sub-paths) that are shared with other vehicles are at the macro level. The authors propose a coordination system in which each agent plans its motion in the following cyclic steps:

1. Check for shared nodes on the path.
2. Communication with neighbours.
3. Priority group creation.
4. Check Macro/Micro resources shared with neighbours.
5. Competition for resources.
6. Access and speed management.
7. Use of resources and update state.

Their algorithm is proven to avoid both collisions and deadlocks. Moreover, computation time does not increase when the number of vehicles increases.

# 3 Problem Description

The problem considered in this thesis is to route a fleet of AGV's such that transshipment requests are answered as quickly as possible, resulting in high quay crane productivity. In order to do this, deadlocks and collisions must be avoided. A deadlock occurs when two or more vehicles are allocated the same area, so that none is able to continue its route and the system is blocked. Livelock is generally used to describe situations where an AGV is blocked repeatedly by other AGV's without having the possibility to drive to the area which is next on its route (Möhring et al., 2005). Although not as catastrophic for performance as a deadlock, livelocks should be avoided as much as possible. Collision occurs when two vehicles are simultaneously occupying the same space.

The terminal is represented as a graph $G = (V, A)$, where $V$ is the set of vertices, and $A$ is the set of arcs. Vertices represent points at which loading and unloading of AGV's takes place, but also arbitrary points on the terminal's road lanes. A road lane $h \in H$ is defined as an alternating sequence of nodes and arcs $i_1, a_1, i_2, ..., i_n$, where all vertices have the same x-coordinate. The set of buffers $B \subset V$ contains locations at which vehicles may wait without interrupting traffic on the road lanes. Let $S(i, j)$ be the set of arcs that are blocked when the center of an AGV is on arc $(i, j)$. If (the center of) an AGV traverses the arc $(i, j)$ in a certain time period, no other AGV may be present on any arc $u \in S(i, j)$. This is to prevent costly collisions. Before operations start, every vehicle is positioned at its starting location, which is always a buffer.

Arcs have their distance given by $d_{i,j}$, with $i, j \in V$ and $(i, j) \in A$. Travel times depend on the speed of an AGV while travelling over the particular arc. The maximum speed on arc $(i, j)$ is given by $v_{i,j}^{max}$. Let the maximum acceleration (deceleration) of AGV $k \in K$ be $a_k$ $(b_k)$ per time unit.

This thesis considers the real-time version of the AGV routing problem. Consequently, the requests occur in an unknown sequence and each request is assigned to a single AGV, which is then routed without any knowledge of requests that will arrive later. A transshipment request $p = (o_p, d_p, \theta_p)$ contains an origin node $o_p \in V$, an destination node $d_p \in V$, and a desired starting time $\theta_p$. Without loss of generality, $\theta_p$ is assumed to always be the time at which the request comes in. Requests arrive according to a stochastic process $SP$.

## 3.1 Mathematical Formulation

Suppose a new request comes in for an AGV. At this point, other vehicles may already be satisfying their requests. If a vehicle $k$ is already busy satisfying request $p \in P$, then $o_p$ is set to the current position of $k$. Let $R$ be the set of feasible routes, with $a_{rp} = 1$ if route $r$ contains a feasible path from $o_p$ to $d_p$, and zero otherwise. Let $b_{rijt} = 1$ if and only if route $r$ has a vehicle driving on arc $(i, j) \in A$ at time $t$. The total completion time of route $r$ is equal to $c_r$. Decision variable $x_r$ is 1 if route $r$ is used, and zero otherwise. Using the variables and parameters discussed so far, a mathematical formulation for the dynamic AGV routing problem is given below.

$$\min \sum_{r \in R} c_r x_r \tag{1}$$

$$\text{s.t.} \quad \sum_{r \in R} a_{rp} x_r = 1 \qquad p \in P \tag{2}$$

$$\sum_{r \in R} \sum_{(u,v) \in S(i,j)} b_{ruvt} x_r \leq 1, \qquad (i,j) \in A, t = 1, ..., T \tag{3}$$

$$\sum_{r \in R} x_r \leq |K|, \tag{4}$$

$$x_r \in \{0, 1\}, \qquad r \in R \tag{5}$$

The objective (1) is to minimize the sum of completion times of all current requests, including the new incoming one. The faster requests are completed, the higher QC productivity will be in terms of containers moved per hour. Constraint (2) ensures that all requests are satisfied and (3) prevents collisions by allowing at most one AGV per arc $(i, j)$ and all arcs in $S(i, j)$. Finally, constraint (4) makes sure that the number of routes used does not exceed the AGV fleet size.

A feasible route $r$ is a path in the graph $G' = (V', A')$ that starts each request at its origin and delivers the container to the destination node before picking up a container for the next request. A vertex $i \in V'$ in this graph includes both a node on the graph representing the terminal and the velocity with which the vehicle arrives at this node. In a feasible path, the velocity of an AGV should equal zero at the first and last node. Allowed changes in speed between two nodes depend on the acceleration and deceleration capabilities of an AGV, which depend on the load it carries on its particular request. Therefore if $a_{rp}$ equals 1, route $r$ contains a path from $o_p$ to $d_p$ in $G'$ that satisfies the acceleration and deceleration restrictions imposed by the load of request $p$.

Note that the number of feasible routes grows exponentially with the number of nodes in $G$, and the the number grows even faster when considering all possible velocities. Column generation and other techniques may be used to reduce the number of routes considered

## 3.2 Differences with a traditional VRP

The proposed problem can be regarded as a pickup and delivery problem in which all vehicles have a capacity of one container. This is a variation of the Vehicle Routing Problem (VRP), which is a very well known problem in the literature. However, there are some important distinctions that restrict the use of standard methodology (Qiu, Hsu, Huang, & Wang, 2002).

First of all, a VRP typically considers a path network on a much larger scale than a container terminal. As a result, dimensions of the vehicles become insignificant and vehicles can be regarded as moving points. Also, collisions and congestion are ignored. For an AGV system, on the other hand, request distances are short and the space that an AGV occupies cannot be disregarded. In addition, collisions may occur and congestion will be observed when too many AGV's are driving in one area simultaneously, so that some have to wait for others to pass.

Second of all, requests in a static VRP are ready to start and known from the beginning. The online AGV system considered in this thesis is real-time, in the sense that requests are unknown and may only be executed once they have become known.

Furthermore, the shortest distance path typically coincides with the shortest time path for a standard VRP. Consequently, a path for a vehicle is uniquely defined by its start and end node. This is not true in general for an AGV system, because the shortest distance path may be congested.

Finally, apart from the decision of which arcs should be used by which vehicle, velocity also becomes a decision variable in the AGV system optimisation. As a result, the set of routes $R$ grows very rapidly with the number of arcs in the graph, even when velocity is discretised. Standard VRP solution methodology such as column generation is therefore not recommended for large instances such as a container terminal.

## 3.3 Complexity

Although formulated mathematically as a VRP, the described problem is typically categorised as a Multiple Agent Routing Problem (MARP). This thesis also considers speed change limitations, which can be regarded as additional constraints. By a reduction from the strongly NP-hard Flow Shop Problem with Blocking, it can be shown that even without additional side-constraints, the MARP is an NP-hard problem (Ter Mors, 2009).

# 4 Request Assignment and Routing

Although this thesis focuses on driving, an incoming transportation request needs to be assigned to a vehicle and routed from its origin to the destination of the request. Request assignment and routing will be discussed shortly, because methods for both are required for running simulation experiments.

## 4.1 Request Assignment

The request assignment (dispatching) method used is now addressed quickly, as it may influence simulation output. Dispatching is done by the central computer, which can assign an incoming request to all AGV's that are not completing a request at the time. An exception to this rule is made when all AGV's are busy. None of the vehicles is dedicated to a subset of QC's. Instead, the AGV's are pooled, such that each vehicle can work for all available cranes. Given a list of available AGV's, a request $p = (o_p, d_p, \theta_p)$ is assigned to that vehicle $k \in K$ which scores highest based on the following variables:

- Travel distance from the current location of $k$ to $o_p$, where shorter distances are preferred
- Urgency of the request, measured by how much slack there is between the estimated earliest arrival time of $k$ and the latest time at which $k$ has to arrive at $o_p$ in order to arrive at $d_p$ in time. More slack is preferred.
- Vehicles in the proximity of $k$, where more vehicles are preferred.

After a request has been assigned to a vehicle, a route from $o_p$ to $d_p$ has to be made. This is discussed in the next subsection.

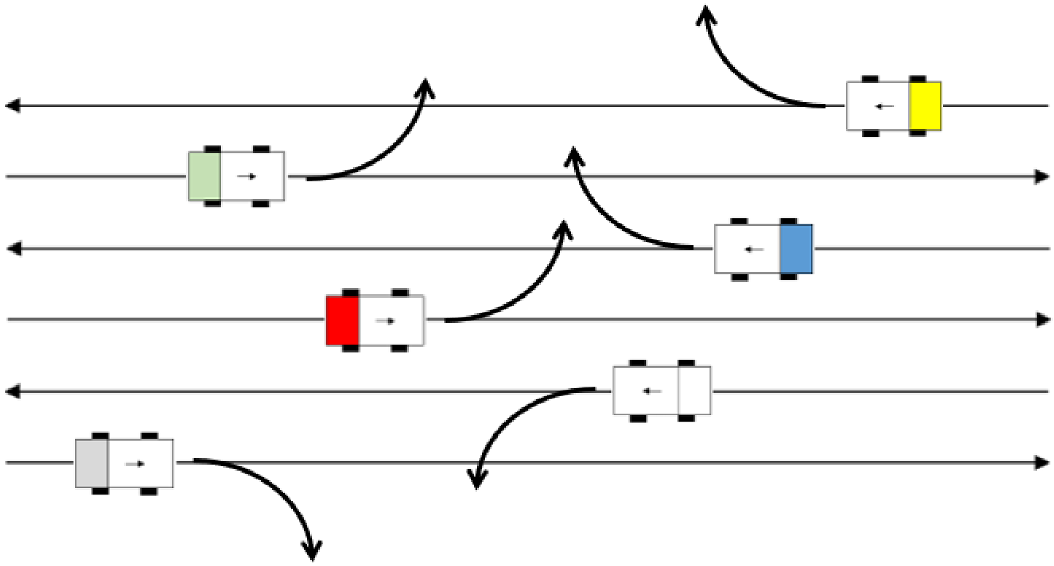
## 4.2 Routing

In order to transport containers by AGV's from origin to destination efficiently and without collisions, a separated method is used. First, a route is determined to satisfy the request, after which the AGV's are responsible for avoiding collisions and conflicts when actually driving their predetermined routes. This subsection discusses the central routing approach that is used to create a route for each request that is assigned to an AGV.

The routing method proposed is mainly an organisation strategy of the highway and quay lanes. With the organisation in place, route determination is rather simple. The first two lanes (from the top) of the highway are reserved for vehicles travelling from one transfer point under a quay crane to another. Lanes three and four are for AGV's driving from a yard stack module to the quay. The last two lanes are dedicated to all vehicles that are headed for the yard. An illustration of this organisation is provided in Figure 1.

Organisation of the quay is rather straightforward. All four lanes are unidirectional in the same direction, so that once again a deadlock is made impossible between vehicles making turns from opposite directions.

Figure 1: Organisation of the highway



This figure shows the distribution of lanes on the highway. Buffers and quay lanes are located above the highway and the yard and its stacking cranes are below the highway.

A route for a request $p = (o_p, d_p, \theta_p)$ can now be created quickly by finding the shortest distance path from $o_p$ to $d_p$, for example by using Dijkstra's algorithm. Because the imposed lane restriction must be respected, route planning boils down to picking an appropriate lane, and determining where to enter and exit the lane. This last decision is also rather easy, since buffers and loading and unloading points are right next to the highway and/or quay lanes. Implementation of this routing strategy is readily available in TBA's TIMESquare simulation software. Throughout many simulations, it has shown to prevent most deadlocks in the routing process already. This is achieved mainly because the division of lanes is such that vehicles can never turn into each other.

# 5 Driving

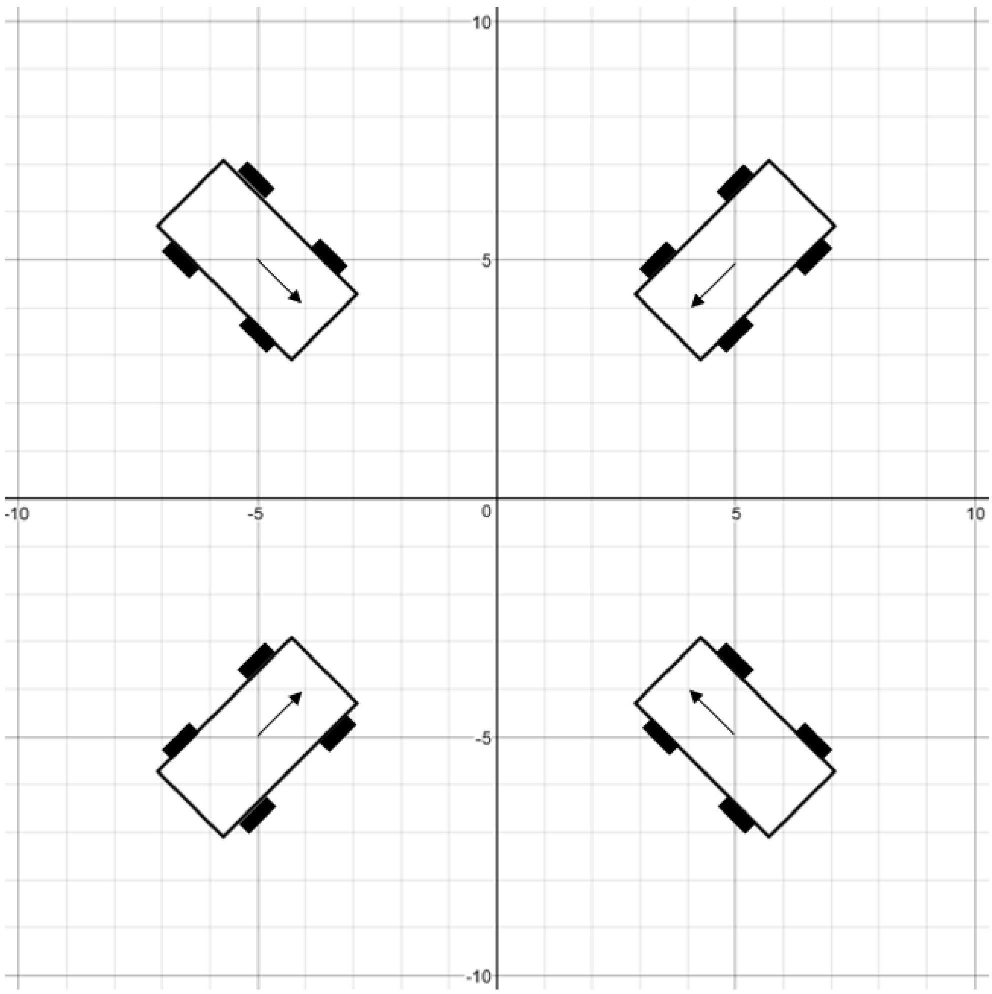
This section presents the main methodology used for driving and coordinating the AGV's. Both TBA's centralised strategy and a new autonomous approach are discussed.

The position of AGV $k$ at time $t$ is given by $z_{kt} = (x, y, \gamma)$. With the terminal modeled as a Cartesian plane, $x$ and $y$ are respectively the horizontal and vertical coordinates of $k$. Because an AGV is not a single point, rotation coordinate $\gamma$ around the x-axis is required to model the complete space that the vehicle occupies. The value for $\gamma$ is always between 0 and $2\pi$ (radians), so that the direction of $k$ is also defined. Figure 2 shows the position of four example AGV's. The coordinates $(x_0, y_0)$ of the center-front of a vehicle with length $l$, width $w$ and coordinates $(x, y, \gamma)$ are given by:

$$x_0 = x + \frac{1}{2}l \cdot cos(\gamma) \tag{6}$$

$$y_0 = y + \frac{1}{2}l \cdot sin(\gamma) \tag{7}$$

Figure 2: Positions of four AGV's




Four AGV's on a Cartesian plain. The positions are as follows: top-left = (-5, 5, $\frac{7}{4}\pi$), top-right = (5, 5, $\frac{5}{4}\pi$), bottom-left = (-5, -5, $\frac{1}{4}\pi$), bottom-right = (5, -5, $\frac{3}{4}\pi$).

For collision avoidance, one should be most interested in the corner coordinates of AGV's. The Cartesian coordinates of the right-front, left-front, right-rear and left-rear corner and respectively given by $(x_{rf}, y_{rf})$, $(x_{lf}, y_{lf})$, $(x_{rr}, y_{rr})$ and $(x_{lr}, y_{lr})$. Expressions for these coordinates are as follows:

$$x_{rf} = x + \frac{1}{2}l \cdot cos(\gamma) + \frac{1}{2}w \cdot sin(\gamma) \tag{8}$$

$$y_{rf} = y + \frac{1}{2}l \cdot sin(\gamma) - \frac{1}{2}w \cdot cos(\gamma) \tag{9}$$

$$x_{lf} = x + \frac{1}{2}l \cdot cos(\gamma) - \frac{1}{2}w \cdot sin(\gamma) \tag{10}$$

$$y_{lf} = y + \frac{1}{2}l \cdot sin(\gamma) + \frac{1}{2}w \cdot cos(\gamma) \tag{11}$$

$$x_{rr} = x - \frac{1}{2}l \cdot cos(\gamma) + \frac{1}{2}w \cdot sin(\gamma) \tag{12}$$

$$y_{rr} = y - \frac{1}{2}l \cdot sin(\gamma) - \frac{1}{2}w \cdot cos(\gamma) \tag{13}$$

$$x_{lr} = x - \frac{1}{2}l \cdot cos(\gamma) + \frac{1}{2}w \cdot sin(\gamma) \tag{14}$$

$$y_{lr} = y - \frac{1}{2}l \cdot sin(\gamma) - \frac{1}{2}w \cdot cos(\gamma) \tag{15}$$

Given a route, each AGV is responsible for making driving decisions such that the route is executed safely without collisions. Without central instructions, an AGV has to drive at the same speed, accelerate, or decelerate when appropriate. Using sensors, other vehicles can be spotted. AGV's jointly need to complete their routes as quickly as possible, meaning that they should not only care about finishing their own route as quickly as possible. Cooperation is required at intersections, which is enabled through radio communication between surrounding vehicles. The driving problem can be classified as a distributed coordination problem, or more specifically, an autonomous multi-agent cooperative control problem (Carlino, Boyles, & Stone, 2013).

## 5.1 Centralised Driving

This subsection briefly discusses the centralised control strategy currently adopted in TBA' software, the performance of which will be treated as a benchmark. A vehicle satisfying a request $p \in P$ from $o_p$ to $d_p$ needs to ask permission to the central computer for every part of the route it has to drive. Only once a so-called claim has been granted may the AGV enter an area. The system avoids overlapping claims, effectively enforcing AGV's to wait for each other and also preventing collisions. Curves and s-curves cannot be claimed partially and are always part of a series of non-stop claims. These claims are either all granted or all denied. This approach has proven to be very effective in preventing deadlocks, since AGV's cannot come to stand still in the middle of a curve.

Taking into account communication delays and aiming to prevent standing still, AGV's will claim a couple of meters ahead of their braking distance. For centralised drive control, the allowed distance $D$ is the total distance for which a vehicle has granted claims.

The next subsections describe the decentralised strategy proposed and analysed in this thesis. First, speed control is discussed, after which a protocol for collision avoidance is introduced.

## 5.2 Speed Control

Suppose an AGV with load $j \in J$ has to drive a straight trajectory with a total distance of $D$ metres, which coincides with the allowed distance. Note that in a decentralised setting, the allowed distance can be defined as the distance to the closest vehicle in front of the AGV. Furthermore, the starting speed is $v_0$ and the required end speed $v_1$. Finally, let $v_j^{max}$, $a_j$ and $b_j$ respectively denote the maximum speed (m/s), acceleration and deceleration (m/s$^2$) of an AGV with load $j$. How can this trajectory be travelled as quickly as possible, while respecting the starting and ending velocity?

Let $v^*$ be the highest velocity reached on the trajectory. The minimum distance traveled before reaching $v^*$ from $v_0$ is equal to:

$$\int_0^{\frac{v^*-v_0}{a_j}} v_0 + a_j t \ dt = \frac{v_0(v^* - v_0)}{a_j} + \frac{(v^* - v_0)^2}{2a_j} \tag{16}$$

The distance needed to decelerate from $v^*$ to $v_1$ equals:

$$\int_0^{\frac{v^*-v_1}{b_j}} v^* - b_j t \ dt = \frac{v^*(v^* - v_1)}{b_j} - \frac{(v^* - v_1)^2}{2b_j} \tag{17}$$

Note that (17) is equal to the breaking distance if $v_1 = 0$. To minimise travel time on the trajectory, (16) and (17) are combined to look for $v^*$ such that:

$$\frac{v_0(v^* - v_0)}{a_j} + \frac{(v^* - v_0)^2}{2a_j} + \frac{v^*(v^* - v_1)}{b_j} - \frac{(v^* - v_1)^2}{2b_j} = D \tag{18}$$

Solving this equation, the minimal travel time is reached when:

$$v^* = \min\left\{ v_j^{max}, \sqrt{\frac{b_j v_0^2 + a_j v_1^2 + 2a_j b_j D}{a_j + b_j}} \right\} \tag{19}$$

Note that a valid solution should satisfy $v^* \geq v_0$ and $v^* \geq v_1$. If a (valid) solution does not exist, that means it is impossible to accelerate/brake from $v_0$ to $v_1$ before travelling $D$ metres. The travel time corresponding to the optimal $v^*$ can be computed as follows. The distance that the AGV travels at a constant speed, given $v^*$, is equal to:

$$D^* = D - \frac{v_0(v^* - v_0)}{a_j} - \frac{(v^* - v_0)^2}{2a_j} - \frac{v^*(v^* - v_1)}{b_j} + \frac{(v^* - v_1)^2}{2b_j} \tag{20}$$

Note that $D^* > 0$ only if $v^* = v_j^{max}$. The shortest travel time is now calculated as:

$$t^* = \frac{v^* - v_0}{a_j} + \frac{v^* - v_1}{b_j} + \frac{D^*}{v^*} \tag{21}$$

## 5.3 Collision Avoidance with Decentralised Driving

In order to avoid collisions without central control, the AGV's are equipped with sensors, as well as communication hardware used to claim an intersection area (non-horizontal transit). A number of situations will now be discussed in which two AGV's enter each others sensors and actions may be required by one or both vehicles.

### 5.3.1 Same direction, same lane

Suppose two vehicles with the same y-coordinate are travelling in the same direction with different speeds, as is shown in Figure 3. If the red AGV travels with lower speed than the green one, it will be noticed as soon as it enters the sensors of green. At this point, let the speeds of the green and the red AGV be $v_g$ and $v_r$ (m/s), respectively. The acceleration (deceleration) of the green AGV is $a_g$ ($b_g$) and the deceleration of red is $b_r$. Finally, the distance between the two vehicles is $d_{gr}$.

Figure 3: Head to tail collision

To avoid a collision, the following approach is taken. Let $t_{gr} = \frac{d_{gr}}{v_g}$ be the time it takes green to travel the distance to red at its current speed. The distance that red travels during $t_{gr}$ if it would start braking immediately is equal to:

$$d_r(t_{gr}) = min\left[\frac{(v_r - 0)^2}{2b_r},\ \int_0^{t_{gr}} v_r - b_r t\ \ dt\right] = min\left[\frac{v_r^2}{2b_r},\ v_r t_{gr} - \frac{1}{2} b_r t_{gr}^2\right], \tag{22}$$

where the fist term inside min operator corresponds to the braking distance of red at its current speed $v_r$. The so called allowed distance for green will be set to:

$$D_g = d_{gr} + d_r(t_{gr}) - M, \tag{23}$$

where $M$ is the minimum distance margin between two vehicles. This allowed distance is then inputted as $D$ in the formulas for speed optimisation from Section 5.2. Furthermore, $v_0 = v_g, v_1 = 0, a_j = a_g$ and $b_j = b_g$.

Note that the previous calculation of allowed distance is not only performed when the red AGV enters the sensors of green. To the contrary, information is updated at least once every second. This way, speed control and collision avoidance cooperate to work both safely and efficiently.

#### 5.3.2 Opposite direction, same lane

Suppose two vehicles with the same y-coordinate are travelling in opposite direction, with possibly different speeds. This situation is shown in Figure 4.

Figure 4: Head-on collision

This situation is excluded, because only unidirectional lanes are considered. It may happen that when a container needs to be rotated 180 degrees, an AGV drives onto a lane in the wrong direction when leaving the yard, and then changes its direction so that the container now faces the right way. This case, however, is included in Sections 5.3.5 and 5.3.6.

#### 5.3.3 Same direction, nearby lane

Suppose two vehicles with similar x-coordinates are travelling in neighbouring lanes in the same direction, with possibly different speeds. This situation is shown in Figure 5.

Figure 5: Parallel passing

No action is required by any of the two vehicles in order to avoid a collision, even though they may be driving close to each other. As long as both AGV's keep their lane, they will not collide.

#### 5.3.4 Opposite direction, nearby lane

Suppose two vehicles with similar x-coordinates are travelling in neighbouring lanes in opposite direction, with possibly different speeds. This situation is shown in Figure 6

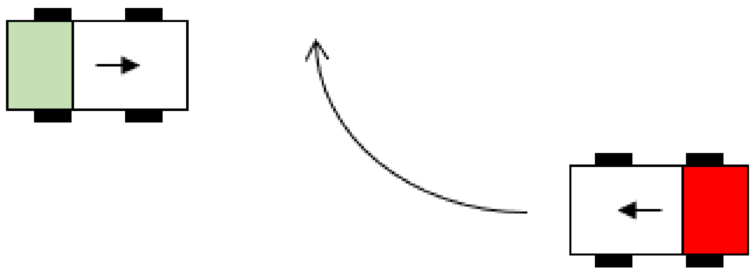
Figure 6: Opposite passing

No action is required by any of the two vehicles in order to avoid a collision, even though they may pass each other closely. As long as both AGV keep their lane, they will not collide.

### 5.3.5 One vehicle travels non-horizontally

Suppose one AGV (red) needs to cross the lane that another AGV (green) is currently occupying. When a vehicle needs to drive a part of its route such that its rotation coordinate $\gamma$ is neither 0 nor $\pi$, it crosses lanes and may collide with another AGV that is not anticipating this. This situation is shown in Figure 7. An example of such non-horizontal travel occurs when a vehicle turns from a buffer onto a lane, or vice versa.

Figure 7: Non horizontal travel

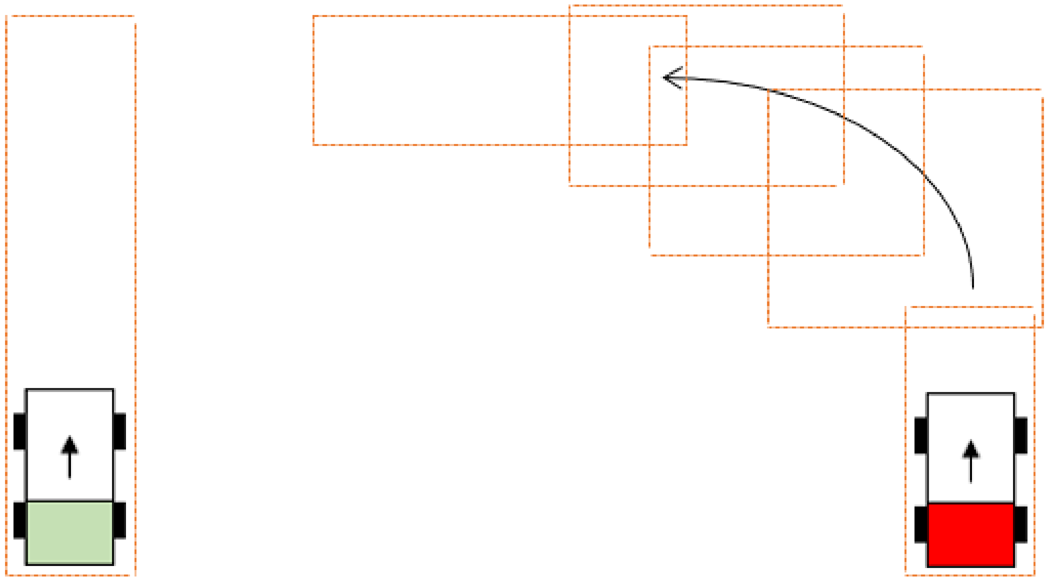


To prevent collisions when a vehicle travels non-horizontally, the complete area during which a rotation coefficient of the AGV is not 0 or $\pi$ needs to be claimed before the vehicle can enter this area. Two types of claims can be distinguished: rectangular claims and curve claims. Rectangular claims are used for straight parts of an AGV's non-horizontal travel. They are exactly long and wide enough to cover the area that the AGV will cover. Curve claims consist of a set of overlapping rectangular claims that jointly cover the area that the vehicle needs to make its curve. The difference between a rectangular and a curve claim is illustrated in Figure 8.

The area of a claim, its start time and (expected) end time are communicated to surrounding AGV's by radio signal. As the claimed trajectory is driven, the claim disappears piece by piece, which is also communicated. The following communication approach is taken. After a claiming vehicle has made a claim, it stays alert to responses from other AGV's. As soon as an AGV (green in Figure 7) is notified about a claim, it checks whether the claim intersects its path. If so, it will adjust its speed such that it can stop before entering the claimed area. If it is not possible for the horizontal AGV to stop before entering the claimed area after the claim's time interval starts, it will not brake and leave the claimed area as soon as possible. This AGV will send a message to the claiming vehicle (red), including its current coordinates and the expected time at which it will have left the claimed area. Once it has left the claim, the claiming vehicles is notified. Having received feedback from one or multiple AGV's, the claiming vehicle will adjust the expected end time of the claim interval. This process of feedback continues until the claiming vehicle no longer receives responses and driving the claim can begin.

For safety reasons, a claim needs to be communicated a minimum number of seconds before the claiming vehicle plans to enter the area. Let $v_r$ and $b_r$ respectively be the velocity and deceleration

Figure 8: Rectangular and curve claims



Two examples of claims made by AGV's. Claimed areas are spaces within the orange rectangles. The green (left) vehicle is travelling in a straight line and makes a rectangular claim, whereas the red (right) AGV makes a curve claim.

of the claiming AGV. Assuming a communication time of $t_{delay}$ seconds, a claim should be made at least $t^{-}$ seconds in advance, given by:

$$t^{-} = 2t_{delay} + \frac{v_r}{b_r} \tag{24}$$

The second part on the right hand side of the equation is the deceleration time of vehicle $r$ needed to brake from its current speed to standstill. This may be necessary if a horizontally travelling vehicle is blocked when passing through the claimed area.

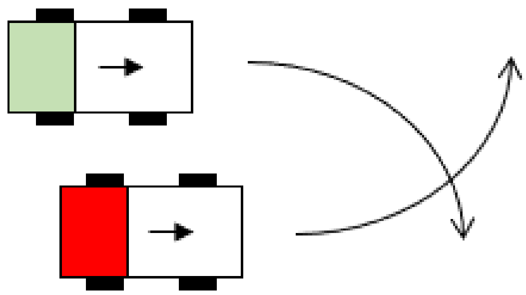
For efficiency reasons, a claiming vehicle wants to prevent having to wait for the claimed area to be cleared. In order to achieve this, red should claim timely, meaning more than $t^{-}$ seconds beforehand. Specifically, red will claim as soon there is no vehicle between itself and the beginning of the area to be claimed, and if it cannot stop anymore for claims made up until the claim area.

There are two main arguments to support giving priority to non-horizontal travel. First of all, communication between vehicles induces small delays, which may become quite substantial when vehicles have to negotiate about priority. Rule based priority makes such negotiations unnecessary. Because vehicles planning to travel non-horizontally have to communicate their claim, it is convenient to give them priority, since this once again leads to less communication. A horizontally travelling vehicles only needs to respond if it cannot stop in time for the claim. If it can stop in time, it should simply do so and no further communication is necessary. Secondly, prioritising non-horizontal travel is an effective way to prevent livelocks. Because a vehicle claiming is allowed to make a reservation claim, it cannot be repeatedly blocked by horizontal AGV's on the highway.

#### 5.3.6 Two vehicles travel non-horizontally, with overlapping claims

Suppose now that two vehicles both need to make a turn, thus travelling non-horizontally, and that their areas to claim (partly) overlap. This scenario is shown in Figure 9.

Figure 9: Overlapping claims



An AGV cannot claim an area that already has been claimed by another vehicle. Furthermore, as stated in Section 5.3.5, the full area over which non-horizontal travel occurs must be claimed. As a result, overlapping claims are treated first come first served. Referring to Figure 9, this means the red AGV will claim and travel first. The green AGV can only travel once red has left its claimed area. The green AGV can already make a "reservation claim" for the area, meaning it claims the area starting immediately after the current expected leaving end time of the red claim. If the red AGV is delayed for some reason, green's claim will automatically shift with the expected leave time.

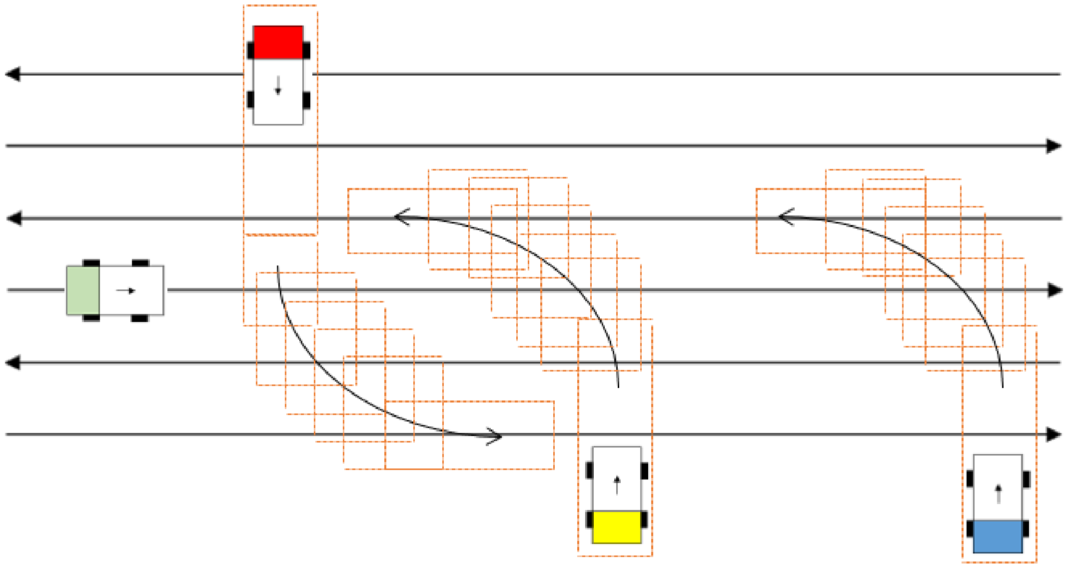
#### 5.3.7 Multiple claims on the same lane

Finally, suppose that an AGV is driving horizontally and faces multiple claims on its lane. Assume the claims are non-overlapping and have different time intervals attached to them. This situation is illustrated in Figure 10. The green vehicle has to decide which claims to cross before stopping. Suppose that green is able to stop in time before entering all claims. Furthermore, it has enough time to cross the areas claimed by red and blue before these claim intervals start, but this does not hold for yellow's claim. Finally, there is not enough room for green to stand still in between the red and yellow claim areas, but this is possible between the yellow's and blue's claims. Which areas are allowed to be crossed at what times by the green AGV?

Because green cannot pass red's area without entering yellow' claim, green may not enter either of these areas, since it cannot leave yellow's area in time and non-horizontal motion is prioritised. Consequently, green has to stop before entering red's area and has to wait until both red and yellow have completed their claims.

As a general rule, the following is proposed. An AGV that travels horizontally towards multiple claims has to stop before the first claim it encounters for which it can stop in time, but that it cannot cross in time. This claim is denoted by $c_{block}$, which would be yellow's claim in Figure 10. The AGV then finds the latest claim before which it can stand still without occupying a claimed area. This claim $c_{stop}$ can be any claim before and including $c_{block}$. In Figure 10, this

Figure 10: Multiple claims



Green faces claims of (in order) red, yellow and blue. There is no space to stand still between the claims of red and yellow.

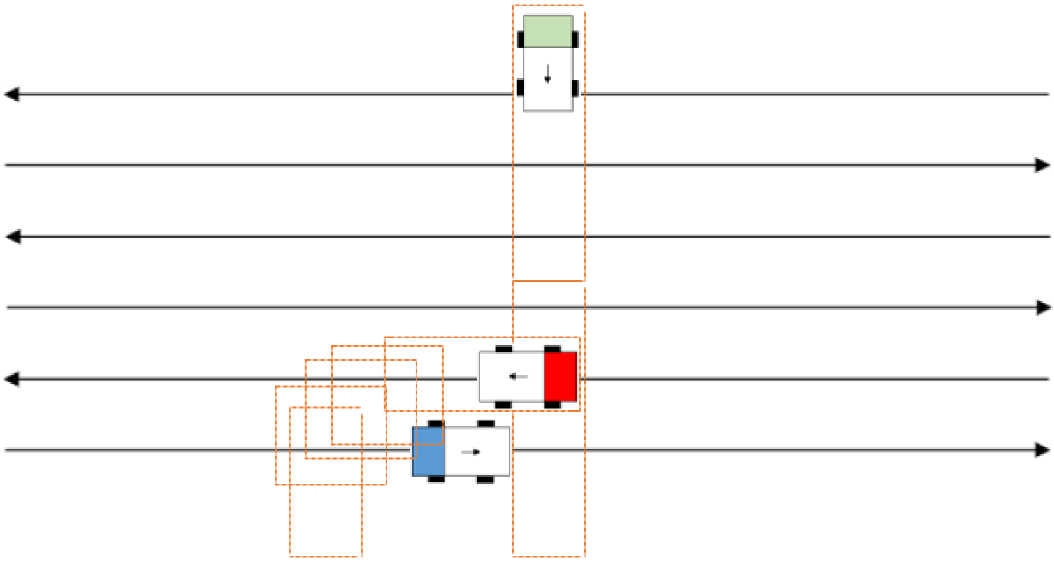
would be red's claim, which is the claim that green will have to stop in front of. If $c_{block}$ does not exist, all claims must be crossed. If $c_{block}$ block does exist but $c_{stop}$ initially does not, then the horizontally travelling AGV must stop before the first claim possible, which is then defined as $c_{stop}$. In this last case, the AGV will stand still in a claimed area, which it needs to communicate to the corresponding claiming vehicle.

#### 5.3.8 Deadlock Prevention and Recovery

Attempts to prevent deadlocks are present in the routing method presented in Section 4.2. In addition, prioritised movements (non-horizontal) are always claimed in full and are non-stop, meaning that a vehicle can only come to a standstill when its rotation coordinate is either 0 or $\pi$. These two measures combined should prevent all deadlocks between two vehicles and most deadlocks between more than two vehicles.

Because of exceptionally unfortunate timing of claims, deadlocks between more than two vehicles may sometime occur. An example of such a deadlock is shown in Figure 11. The green vehicle has made a claim for his vertical travel from a quay buffer across the highway towards the yard. However, it is blocked by the red vehicle, which wants to make a turn from the highway into a yard module. This red AGV in turn cannot proceed because part of its claim is occupied by blue. But since the blue vehicle is travelling horizontally, it cannot cross green's claim. We now have a circular situation: red blocks green, blue blocks red and green blocks blue.

Figure 11: A Deadlock



The obvious resolution here it to allow blue to cross, after which red can make its turn and then green can move as well. Blue has to break a priority rule, so an exception has to be made. The following is proposed: claiming vehicles that are standing still communicate the vehicle that is occupying their current claim. If a claiming vehicle is waiting for a blocking vehicle that is also standing still, then its claims may be ignored by all AGV's. This strategy helps to recover from deadlocks, as can be illustrated by looking at Figure 11. Green communicates it is blocked by red, while red communicates it is standing still because of blue. Now blue knows that green is waiting for a vehicles that is standing still. Therefore, it may ignore green's priority and proceed along its route. The deadlock is solved.

# 6 Simulation Experiments and Results

In order to evaluate the efficiency of the distributed driving approach and to compare it to TBA's original centralised strategy, various simulation experiments are performed. This section describes the experimental setup for simulation and the scenario and terminal used for testing. Results of the experiments are presented, as well as those of sensitivity analysis.

## 6.1 Experimental Setup

This subsection discusses the experimental setup. First, the simulation software of TBA, TIMESquare, is introduced. After that, the terminal used in the simulation model is described in detail.

### 6.1.1 Simulation

TIMESquare is a software package provided by TBA that can model the behaviour of QC's, AGV's, yard cranes, trucks and trains in detail. Using discrete event simulation, an ordered list of events is stored, as well as the time at which each event occurs. An example of such an event is the unloading of a container by a QC, which leads to a request for an AGV. The models include various stochastic values for QC processes, such as movements between transfer points and platforms, hoist and spreader movements, twistlock handling and container grabbing and dropping.

A stochastic value that may particularly impact productivity is the distribution of QC move types. TIMESquare distinguishes three different move types, each of which can either be a load or discharge move. From most to least productive, the move types are: single, tandem and twin. In a tandem move, two containers are picked up simultaneously, long sides adjacent. With a twin lift, two (20-foot) containers are lifted short sides adjacent. The reason that twin lift is (much) more productive than tandem lift is because the two containers lifted by a twin lift can be moved on/from the same AGV. Tandem lifts to and from the vessel by the waterside trolley still need to be moved from and onto AGV's one by one. As a result, tandem moves result in rather small productivity gains compared to single load and discharge, whereas twin lifts can almost double the number of boxes moved per hour. Move type distributions are determined at the start of every simulation run. Proportions of the move types single, tandem and twin in the simulations run for this thesis have ranges 21%-25%, 10%-13% and 62%-69%, respectively.

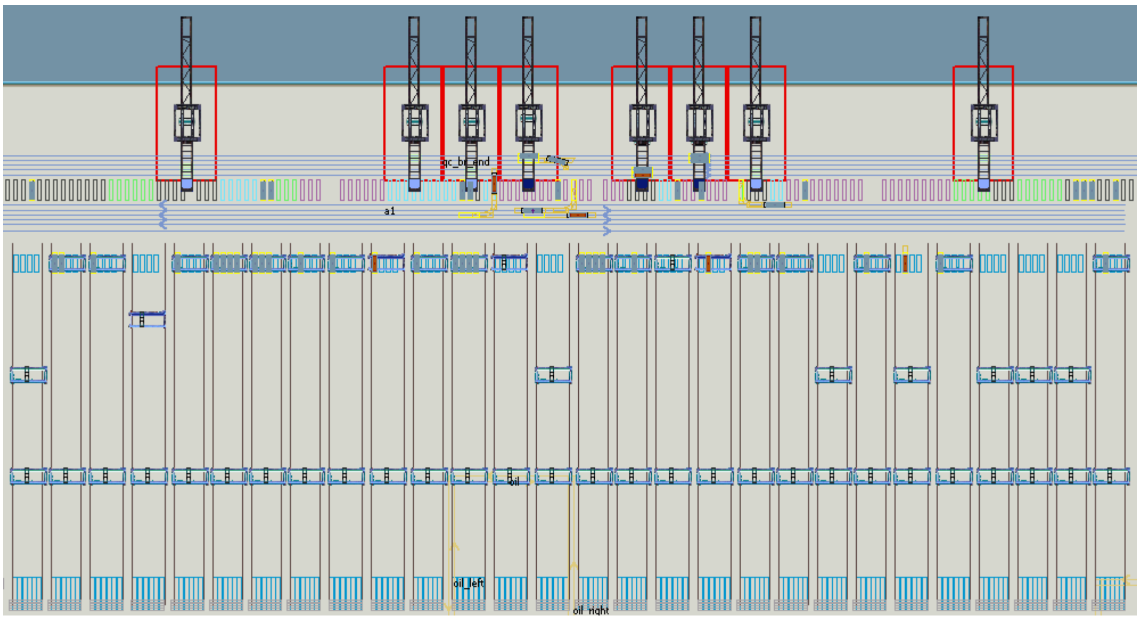
Concerning the AGV's, TIMESquare can model speed and load-based acceleration and deceleration, precise positioning and energy consumption. Communication delays between AGV's and the central computer can be set by the user, even though the simulation itself runs on a single processor.

A single simulation run corresponds to 8 hours of work and processing on the terminal. The first hour of each run is disregarded in order to eliminate the influence of the initial location of the AGV's, as well as initial idleness. Consequently, one replication produces 7 hours of useful data. In general, at least 100 hours of data are required to draw conclusions that are not based on extreme stochastic deviations. The results in this thesis all come from 140 hours of simulation data.

### 6.1.2 Scenario

This subsection describes the scenario terminal on which the proposed driving strategy is tested and compared to the centralised approach adopted by TBA. A total of 8 quay cranes are used on the terminal to load and unload vessels. The yard contains 28 stacking modules, each operated by 2 Rail Mounted Gantries (RMG). There are four unidirectional lanes underneath all quay cranes and 6 additional highway lanes that are parallel to the quay and perpendicular to the yard's stacking modules. Buffers are located between the quay lanes and the highway. All roads on the terminal that can be used by AGV's are equipped with electromagnetic guide paths for the vehicles to follow. Lanes are unidirectional, but a vehicle may initially turn onto a lane in the wrong direction if this is necessary to make a certain curve. The quay lanes all have the same direction, whereas there are (alternately) 3 highways in one direction and 3 in the other. Buffers and yard modules must be entered and left, so around these places, roads are bidirectional. Figure 12 is a screen shot from a TIMESquare simulation that shows the scenario terminal from above. Although this terminal does not have a one-to-one correspondence to a real-life terminal, it is a good representation of what an average RMG terminal looks like.

Figure 12: The Terminal



The terminal used in the simulation as seen from above. From top to bottom, one can see: water, 8 quay cranes, 4 quay lanes, buffers, 6 highway lanes and 28 stacking modules that each contain 2 RMG stacking cranes. Empty AGV's are grey, whereas laden ones are brown.

The AGV's are bidirectional and symmetric. This means that they can move forward and backward in an identical fashion. Further properties of the AGV's are:

- Each AGV has a starting location in a yard buffer.
- An AGV can carry one 40-foot or two 20-foot containers at a time.

- AGV dimensions are 18-by-4 metres.
- An AGV's curve radius is 10 metres.
- An AGV travels at a maximum speed that depends on the trajectory, as well as the weight of its load $j \in J$.
- An empty AGV travels at a maximum straight line speed of 6 meters per second (m/s).
- An empty AGV travels at a maximum curve speed of 3 m/s.
- Acceleration and deceleration depend on the weight of load $j \in J$.
- Acceleration (deceleration) of an empty AGV occurs at 0.94 (1.11) m/s$^2$.

Driving requests for AGV's occur as ordered pairs containing a pickup point and delivery point. Pickup and delivery points are located under a quay crane, in the yard, or at the landside of the terminal. After completing a request, the AGV notifies the central computer and is ordered to drive to a new location, possibly the next pickup point.

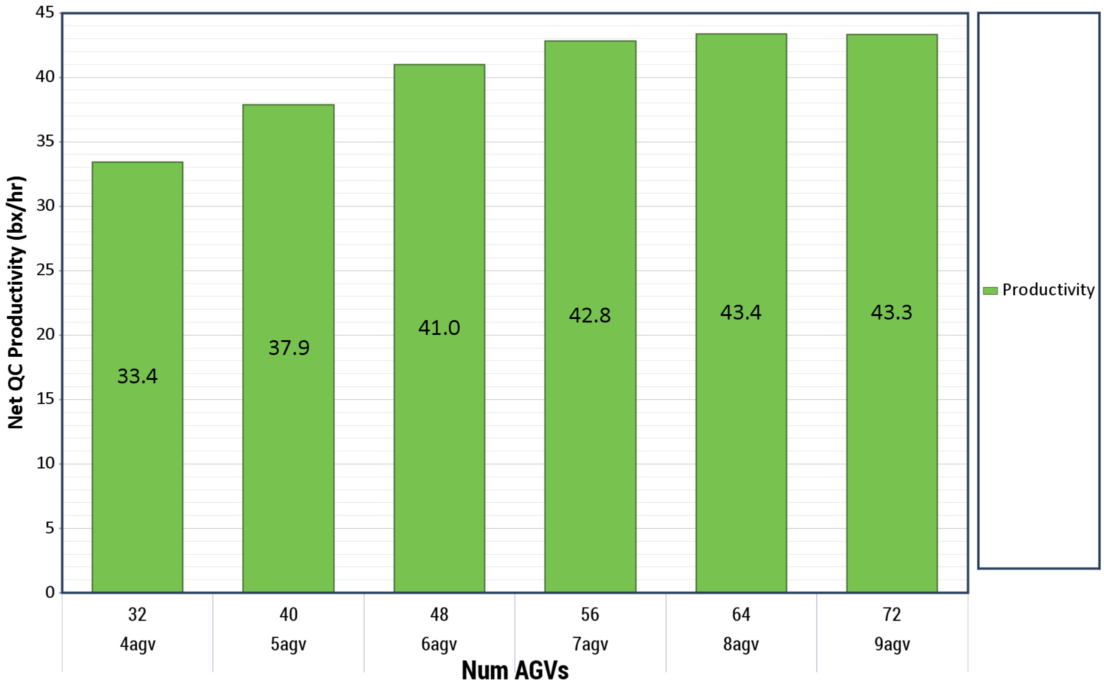
## 6.2 Results

This subsection presents and discusses the results of the simulation experiments performed. First, results of the central driving strategy are discussed. The performance achieved is treated as a benchmark when comparing strategies. Next, the results of the autonomous strategy proposed in Section 5 are analysed. Finally, sensitivity analysis is performed to examine the impact of small changes to the original autonomous methods.

### 6.2.1 Centralised Driving

Figure 13 shows the influence of deploying more AGV's on quay crane productivity, which is measured by the average number of containers (boxes) moved per hour. One can see a clear increase in productivity when the number of vehicles increases from 32 (4 per QC) to 56. Note, however, that the increase in productivity decreases as more AGV's are already present. From 56 vehicles onward, quay crane productivity is rather stable and increasing the number of AGV's deployed does not yield a higher productivity. One explanation for this is that the quay cranes are fully productive and that moving more than 44 boxes per hour is operationally impossible. Another possibility is that productivity it not at its maximum, but using more AGV's leads to congestion and inefficient driving that impedes more containers from being moved.

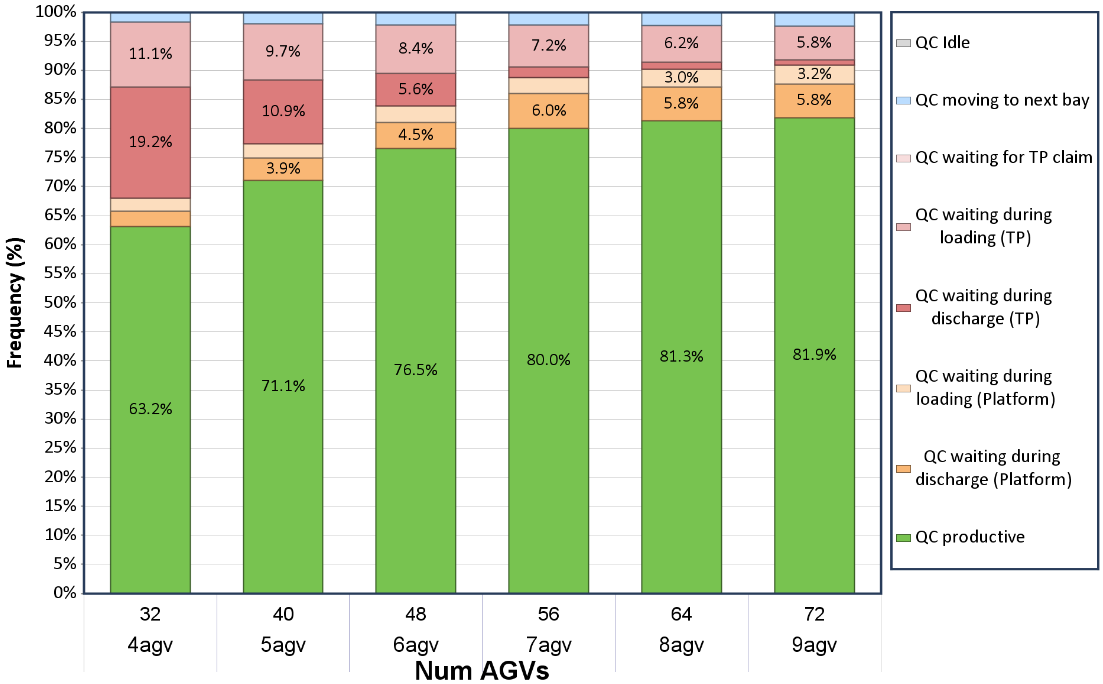
Figure 13: Quay Crane Productivity




Average quay crane productivity, measured by the average number of container moved per hour.

A typical quay crane contains two trolleys, one operating a the landside to move containers between the crane's platform and AGV's, and another that moves containers between this platform and the vessel. Figure 14 shows a decomposition of the occupation time for the land side QC trolley, which is where the interchange with the AGV's occur. It reveals that using more AGV's causes higher quay crane productivity by reducing the time waited for a vehicle to arrive in order to either bring (load) or pick up (discharge). With 4 AGV's per QC, 11.5 minutes per hour are spent waiting on an AGV to pick up a container from the land side trolley. Doubling the number of AGV's reduces this time to less than a minute. Time spent waiting for loading is also reduced significantly with more vehicle deployed, but this difference is less striking. Also notable is that the time spent waiting for the platform actually increases with more AGV's. During loading, this occurs because the the platform is full, whereas during discharge, waiting occurs because the platform is empty and no container can be loaded onto an AGV.

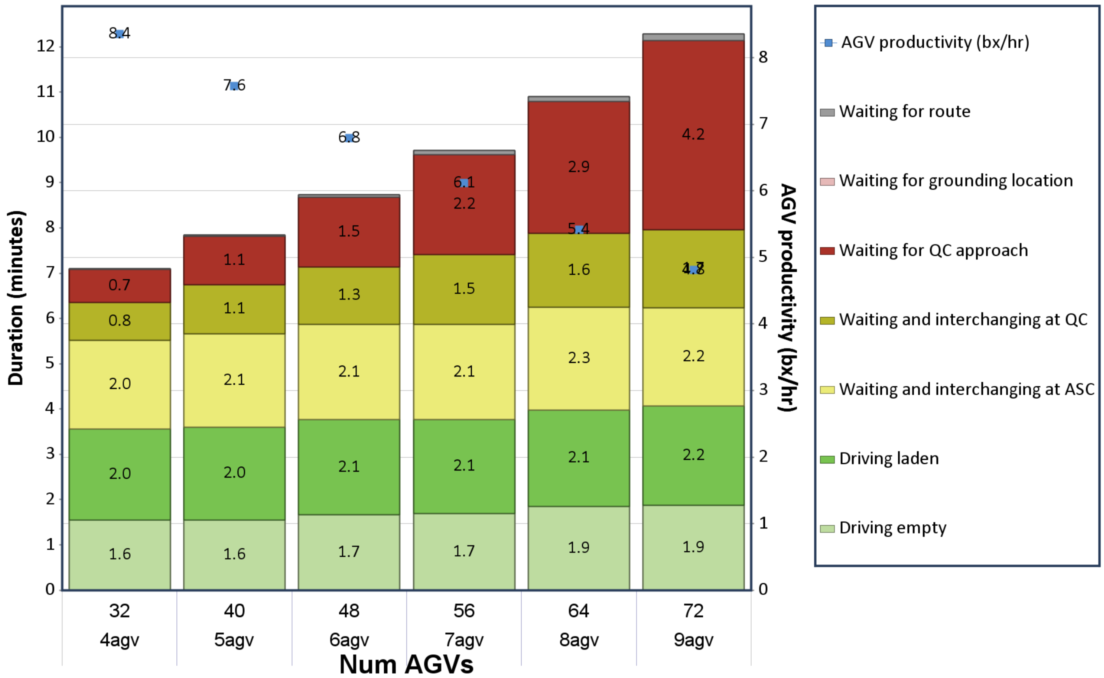
Figure 14: Quay Crane Land Side Trolley




Occupation decomposition of the land side trolley of the QC. This figure gives the percentages of total time during which the quay cranes where occupied with each individual activity.

Moving from the QC to the vehicles, Figure 15 presents the average status of an AGV per container transported, together with average AGV productivity in boxes per hour. Containers moved per hour decrease almost linearly with the number of vehicles in use. The obvious reason for this observation is that when there are (too) few AGV's, they rarely ever have to wait for a QC to (un)load them. On the other hand, when there are many AGV's in use, the quay becomes the bottleneck and AGV's have to wait under a crane quite often, as is indeed shown in Figure 15. The same can be said about the yard, where AGV's have to wait for the Automatic Stacking Cranes (ASC). In addition, because there is little to no congestion with few vehicles, driving to a destination can be done quicker, regardless of whether an AGV is carrying a container.

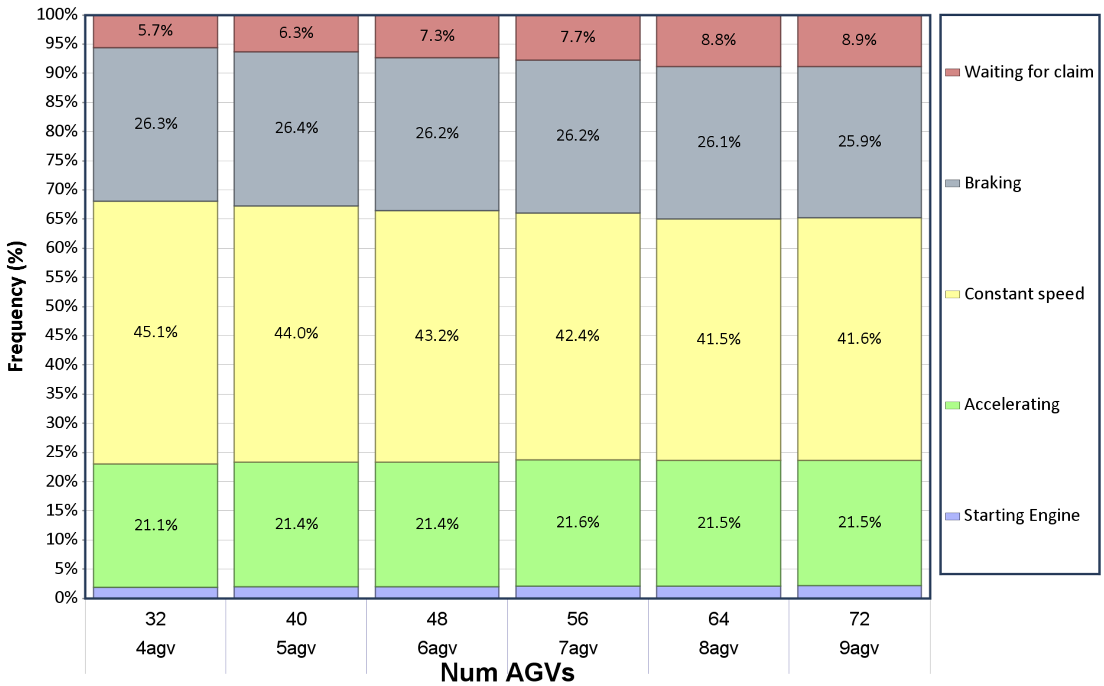
Figure 15: AGV Status per Container




AGV occupation per container, as well as the average number of containers moved per vehicle per hour.

Figure 16 shows the average proportion of AGV drive statuses. As more vehicles are used, congestion increases and vehicles have to wait for each other more often. As a result, vehicles spend more time waiting for a claim. Time spent waiting can obviously not be spent driving, so that in a large fleet, AGV's do not drive at constant speed as much as with fewer AGV's.

Figure 16: AGV Drive Status

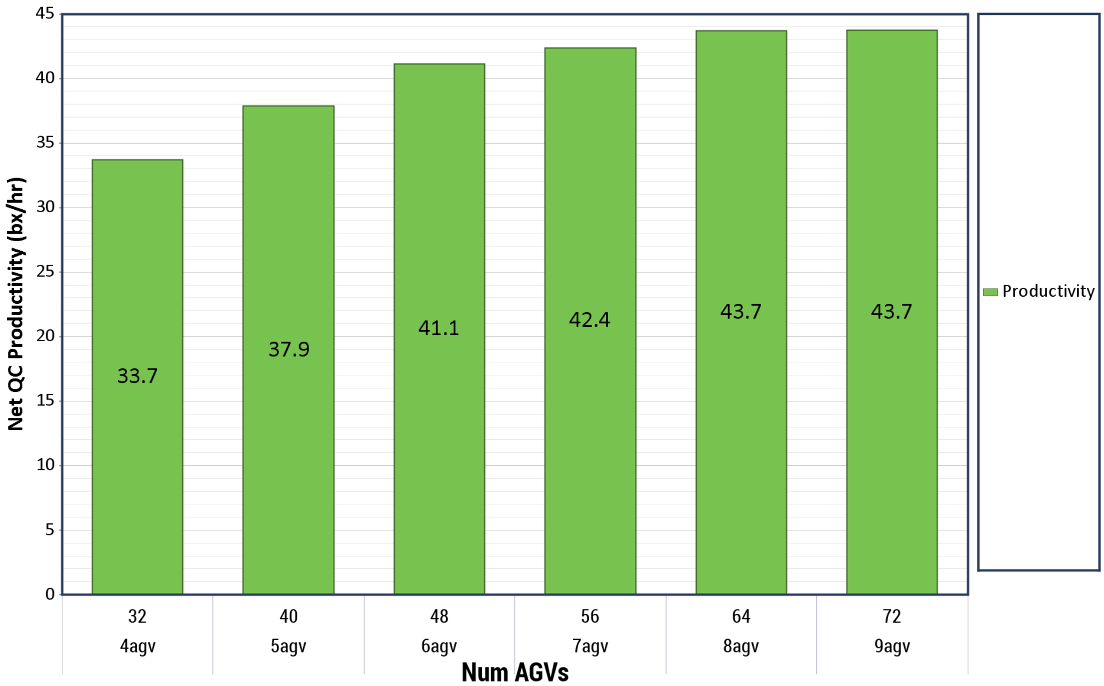



Average AGV drive status as a percentage of the total time in use.

### 6.2.2 Decentralised Driving

Figure 17 shows the average number of containers moved per hour per quay crane for different fleet sizes of AGV's. As with centralised driving, productivity increases as more vehicles are deployed. Comparing the numbers to those in Figure 13, no striking differences in productivity are observed between centralised and decentralised driving. Confidence intervals of 95% can be formed by adding and subtracting half a box per hour from the mean values in these figures. Consequently, for no number of AGV's there is a significant difference in performance between the two strategies.

Figure 17: Quay Crane Productivity (Autonomous)

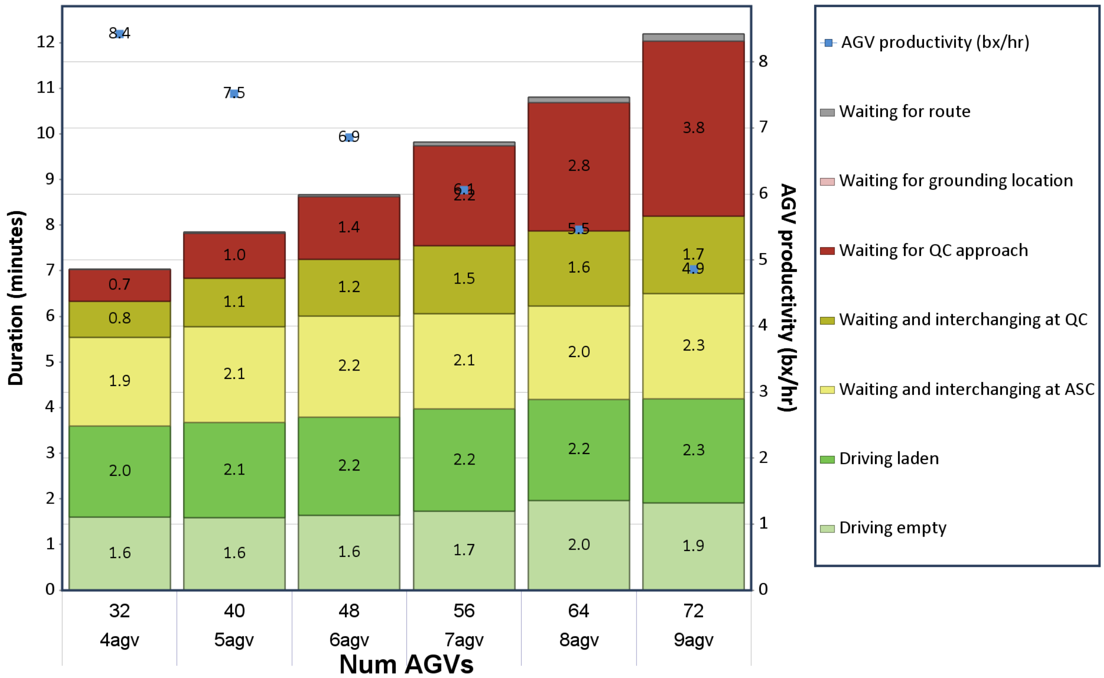



Average quay crane productivity with the autonomous strategy, measured by the average number of container moved per hour.

From an optimisation focused point of view, a centralised strategy seems to have advantages over a distributed approach, because it can use more information to make decisions concerning priority in particular. The fact that the distributed strategy can compete with it very well in terms of operational performance is therefore considered a positive result. This means that if a terminal replaced its centralised strategy with the proposed decentralised approach, it may expect the same performance in terms of containers moved per hour, while unburdening the central computer and making the AGV system robust to failures of vehicles or central communication.

Statistics of the land side QC trolley are very similar for the two competing strategies. However, as shown in Figure 18, the average AGV status per container has some differences. The autonomous approach results in slightly longer driving times (laden and empty) and shorter waiting times at the QC and ASC. It seems that although autonomous AGV's take slightly longer to get to their destination, this only results in shorter waiting times, so no time is wasted and productivity is left unscathed.

Figure 18: AGV Status per Container (Autonomous)




AGV occupation per container, as well as the average number of containers moved per vehicle per hour with the autonomous strategy.

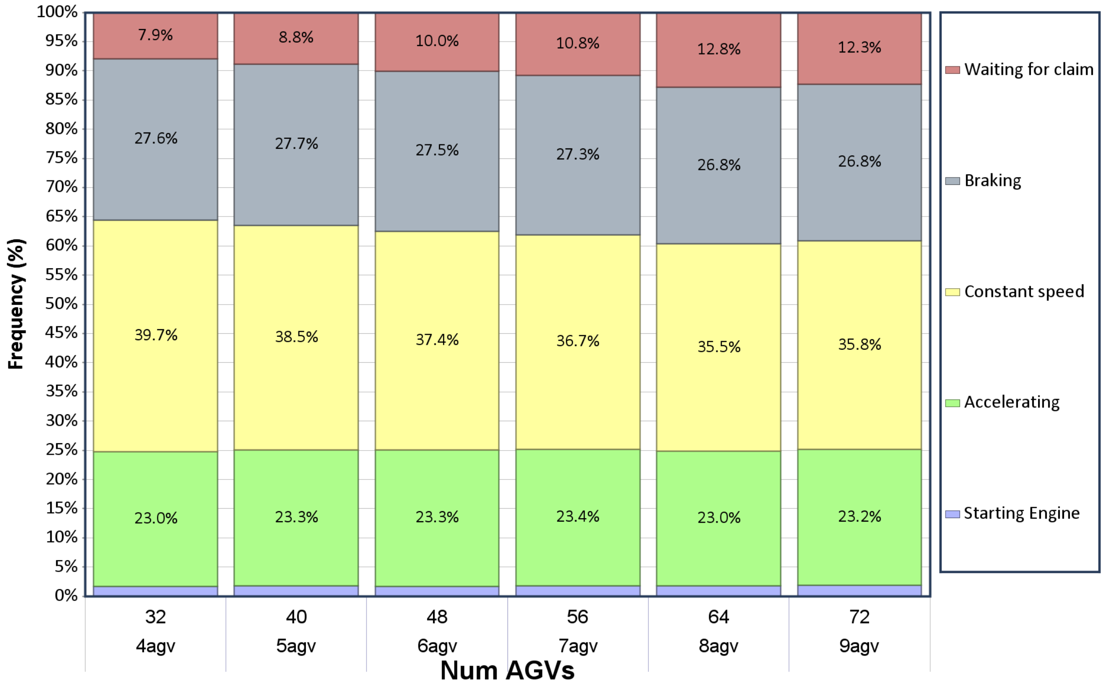
The fact that average driving times are slightly longer in the autonomous strategy can be explained by looking at Figure 19. Comparing this figure to Figure 16, the decentralised approach clearly comes with AGV's adjusting speed more often, resulting in higher proportions for accelerating and decelerating and less driving at constant speed. The reason for this observations is that autonomous AGV's travelling horizontally need to constantly monitor the vehicles in front. If this vehicles is close, many adjustments in speed may be required. On the other hand, in the decentralised strategy, all motion, including horizontal travel, is claimed, so that there are much fewer speed adjustments.

Also, AGV's typically have to wait longer for other vehicles under the autonomous approach compared to when there is central coordination. Although horizontally travelling vehicles do not have to claim and therefore not be waiting for a claim, them standing still waiting for another AGV is still regarded as "waiting for claim" in Figure 19. Because horizontal AGV's also have to wait for claims that are not being driving yet, overall waiting times are on average higher than with centralised driving.

### 6.2.3 Deadlocks and Collisions

In all hours of simulation, not a single deadlock occurred. To detect collisions, the distance between vehicles was monitored at all time. Distances between vehicles were all strictly positive in all simulation runs, meaning no collisions occurred at any time.

Figure 19: AGV Drive Status (Autonomous)




Average AGV drive status as a percentage of the total time in use with the autonomous strategy.

## 6.3 Sensitivity and Variations

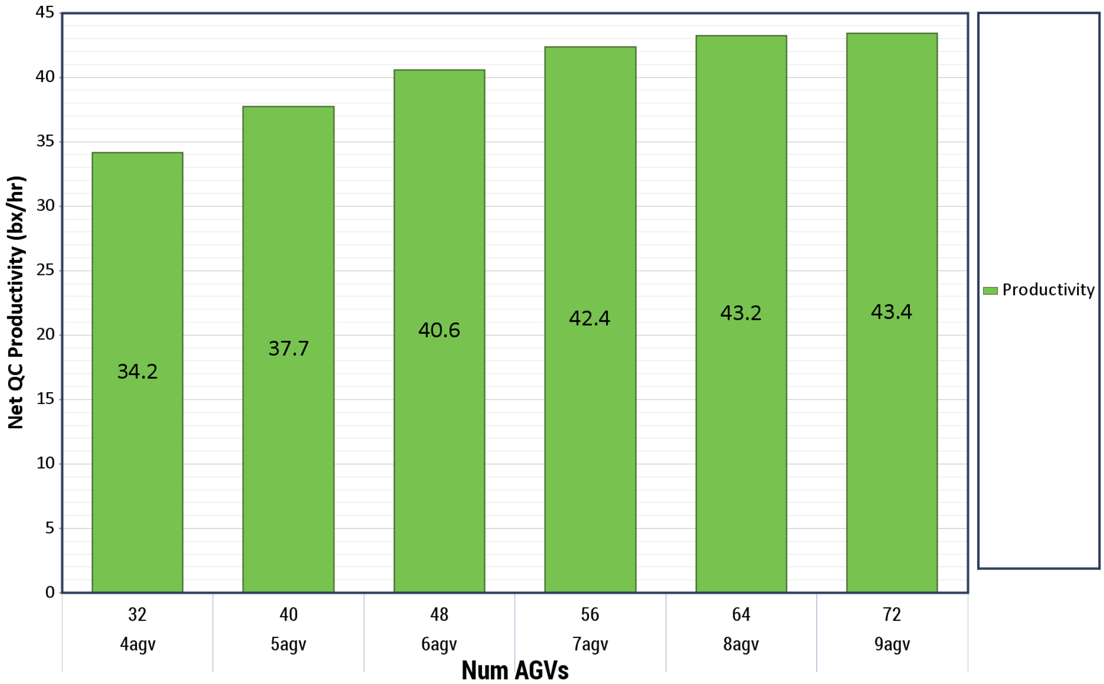
Some variations to the original autonomous strategy are now provided, as well as analyses of the impact of such changes on driving efficiency.

### 6.3.1 Unknown deceleration

In Section 5.3.1, a method was proposed to prevent horizontally travelling vehicles from colliding head to tail. This was achieved by keeping sufficient distance between the two, which is controlled by the trailing vehicle (green in Figure 3). In the calculations used, the maximum deceleration of the vehicle in front (red) is known to be $b_r$. This information would need to be communicated, since it cannot be inferred by simply scanning changes in speed. If deceleration capabilities are unknown, a trailing vehicles would have to assume that $d_r$ is equal to the maximum deceleration of any vehicle with any load. Overall, this would lead to a larger required distance between two horizontally travelling AGV's.

Dropping the assumption of known deceleration force, simulation results can again be obtained. Figure 20 shows the average number of containers moved per hour per QC. Comparing the numbers to those in Figure 13, no clear differences are observed in QC productivity. An explanation for this is that the average increase in distance between two vehicles is typically only a couple of metres, which is too small to make a significant operational difference. Moreover, if the leading vehicle needs to stop or make a turn, the trailing vehicle will have to slow down or possibly even come to a standstill, regardless of whether deceleration is assumed known or unknown. All together, it appears that the assumption of known maximum deceleration is not very strong and that it can

Figure 20: Quay Crane Productivity with Unknown Deceleration




Average quay crane productivity with the autonomous strategy, measured by the average number of containers moved per hour. The maximum deceleration of other vehicles is assumed unknown.

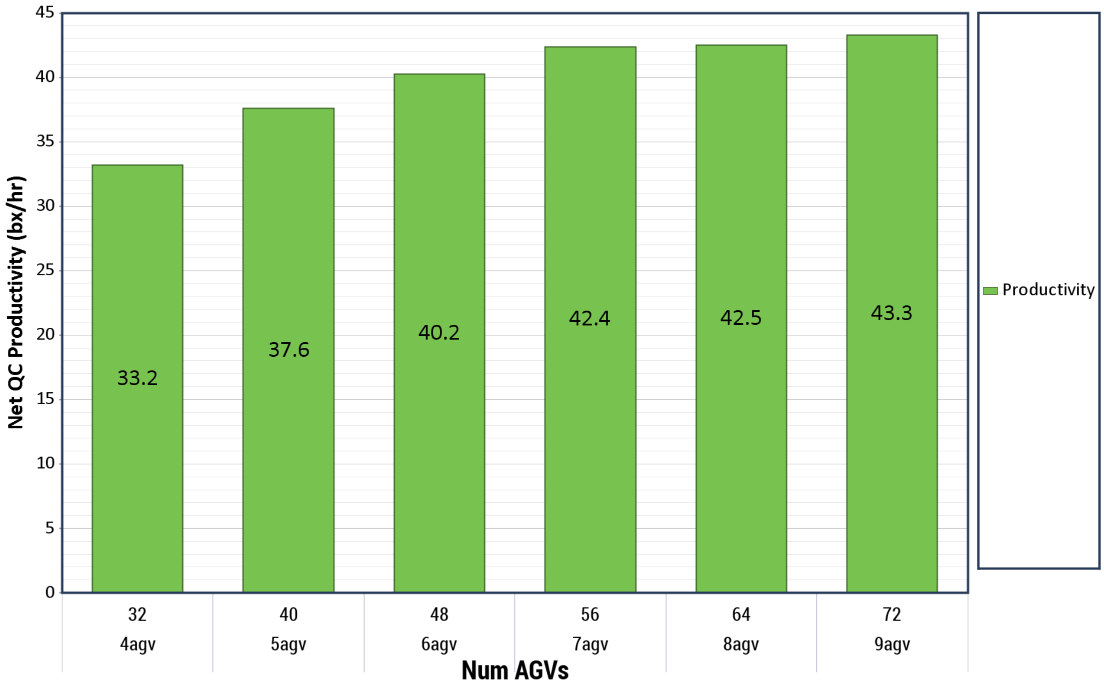
be relaxed without significantly affecting the results.

### 6.3.2 Horizontal Priority

The autonomous strategy employed gives priority to AGV's travelling non-horizontally, meaning horizontal AGV's have to wait for vehicles making a curve or travelling vertically. What if this were to be reversed? Suppose the strategy is changed so that horizontally travelling AGV's are given priority. In this case, non-horizontally travelling AGV's still need to claim their area to drive, but horizontal vehicles do not have to stop for claims if they can. Instead, a horizontal AGV communicates to a claiming AGV if it will cross the claimed area in the corresponding time interval. The claiming vehicle once again needs to update the interval and can only fulfill the claim once no horizontal AGV is hindered by it.

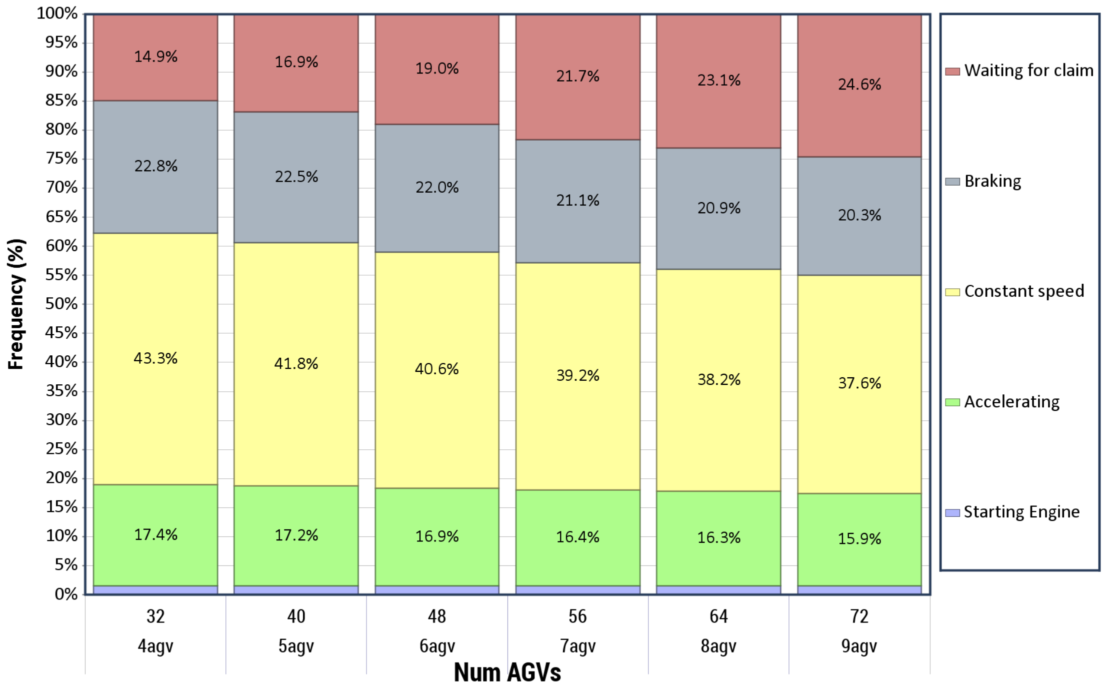
The performance of this strategy in terms of QC productivity is shown in Figure 21. Compared to Figure 17, it shows that horizontal priority results in a lower or equal performance for all fleet sizes. For 6 and 8 AGV's per QC, horizontal priority performs significantly worse than non-horizontal priority. For 7 AGV's per QC, the performance is exactly equal. Finally, for all other fleet sizes, the performance of autonomous AGV's with horizontal priority is worse, but the difference with non-horizontal priority is not significant.

Figure 21: Quay Crane Productivity with Horizontal Priority




Average quay crane productivity of the autonomous strategy with horizontal priority, measured by the average number of container moved per hour.

Figure 22: AGV Drive Status with Horizontal Priority




Average AGV drive status as a percentage of the total time in use with the autonomous strategy. Horizontally travelling vehicles have priority.

Overall, it is reasonable to say that non-horizontal priority is a better choice than horizontal priority. The differences in efficiency become clear when looking at Figure 22 and comparing it to

Figure 19. When non-horizontal AGV's do not have priority, they have to wait until the claim is not hindering a single horizontal vehicles. Since such claims may cross multiple lanes, this can take much time. Livelocks are observed regularly. These occur when a vehicle is repeatedly blocked by other vehicles and does not get a chance to pass (for a long time). As a result, vehicles spend on average almost twice as long "waiting for claim" when horizontal AGV's are prioritised.

# 7 Conclusion

In this thesis, a strategy has been proposed for solving a multi-agent cooperative control problem regarding the transportation of containers within a terminal. Assigning requests and route determination are still done centrally, but driving responsibilities, including collision and deadlock avoidance, are distributed to the individual AGV's. Assuming the AGV's are equipped with sensors and communication technology, a complete driving strategy has been developed. Head to tail collisions between horizontally travelling vehicles are avoided by having trailing vehicles maintain a safe distance to their leading vehicle. Such a safe distance depends on the speed and braking distance of both vehicles. AGV's that (plan to) make a curve or travel vertically need to claim the full area of non-horizontal travel in advance and may only start driving the claim once the complete area is cleared of other vehicles. Horizontal vehicles have to stop for the first claim possible. If they cannot brake quickly enough to avoid entering a claim, this is communicated to the claiming vehicle. The same holds for when a vehicle is blocked and comes to stand still in a claimed area. Regarding overlapping claims, vehicles will travel on a "first claim, first serve" basis. In addition to collision avoidance, deadlocks are prevented by smartly reserving each individual highway lane for a particular purpose when routing. Also, claims are non-stop, so a vehicle can never come to a stand still while travelling non-horizontally. If a deadlock should still occur, a recovery method has been put in place.

To test the proposed strategy and compare it to the centralised driving strategy readily available in TBA's TIMESquare software, discrete event simulation was used. Both strategies were used on different fleet sizes of AGV's on a terminal with 8 QC's that is considered representative of a real-life RMG terminal. Results of the simulation experiments show that the autonomous strategy can adequately compete with the standard centralised approach, with no significant differences in the number of containers moved per QC per hour. This is considered a positive result, since an autonomous strategy can by construction only use local knowledge, whereas a centralised strategy is typically not constrained in this way.

Furthermore, sensitivity analysis has revealed that assuming the deceleration of individual AGV's is common knowledge among all vehicles does not significantly change productivity. On the other hand, reversing the priorities such that horizontally travelling vehicles have precedence negatively affects driving efficiency.

Concluding, let us return to the main research question: to what extent would efficiency of container handling on a terminal change when switching from centralised AGV drive control to an autonomous strategy? The distributed approach taken has all benefits that a distributed approach offers, most importantly offering additional robustness to failures of vehicles and communication, without sacrificing performance compared to TBA's centralised approach. Also, because driving decisions are made by the AGV's themselves, the central computer is greatly unburdened. This means the central computer has additional running time available to make more complex assignments and routing decisions to improve the system's efficiency. A terminal adopting the autonomous strategy will experience an increase in the AGV system's robustness without a reduction in the number of containers it can handle per hour. Of course, this requires the possession of vehicles that are equipped with hardware that allows them to sense and communicate with surrounding vehicles. The terminal may therefore need to make additional investments.

## 7.1 Limitations and Possible Extensions

Finally, some limitations of the methods used are discussed and suggestions are made for future research. First of all, only static priority rules have been considered. In the strategy proposed, non-horizontal vehicles always have priority over horizontally travelling AGV's, disregarding factors such as speed or time since departure. Dynamic precedence rules may become quite complex, but because autonomous vehicles only have to make driving decisions, there is probably enough space for additional computations. An interesting follow up would be to look at the effects of prioritising trains, where a train is informally defined as two or more vehicles travelling close to each other on the same lane. This obviously requires more communication than the original autonomous strategy. Not only do all vehicles in a train need to know they are in, the train as a whole also needs to coordinate with other vehicles to establish priority passing. The advantage of giving priority to trains is that if the front vehicle can continue its path undisturbed, then so can all trailing vehicles. Reversely, if the front vehicle has to wait for a claiming AGV, then so do all vehicles behind it.

Second of all, claiming vehicles now claim an entire area with a single claim interval, resulting in the entire area being blocked. This means that horizontal vehicles may sometimes be able to cross a claim without causing a collision, but are forbidden to do so because the full area is claimed. In an attempt to resolve this, more detailed claiming may be an interesting option. Claiming vehicles could partition their area into parts, each with different, partly overlapping time intervals. This gives horizontal AGV's more flexibility and may save valuable time. Note that the claim should still be non-stop, meaning that all parts of the claim need to be claimed at the same time, such that the time intervals ensure undisturbed transit by the claiming vehicle.

Furthermore, this thesis has considered the methods for creating routes and assigning requests as given. These are the centralised methods present in TBA's software that have been tested to work well for the centralised driving approach. It might be that alternative methods for assignment and routing will work better when adopting the autonomous strategy. However, as the simulation experiments show, performance of the autonomous vehicle fleet is already quite good. Alternative methods for decisions other than driving may result in productivity gains, but changes are expected to be small. For larger gains, it may be necessary to change multiple (possibly all) routes centrally when new request arrives.

Finally, regardless of which initial routing method is used, a promising opportunity for performance gains may be to let vehicles (autonomously) change their routes. This thesis has treated routes as completely fixed to focus on driving these routes as efficiently as possible. Allowing small changes along the way may however help to reduce problems with congestion.

# References


Carlino, D., Boyles, S. D., & Stone, P. (2013). Auction-based autonomous intersection management. In *16th international ieee conference on intelligent transportation systems (itsc 2013)* (pp. 529–534).

de Campos, G. R., Falcone, P., & Sjöberg, J. (2013). Autonomous cooperative driving: a velocity-based negotiation approach for intersection crossing. In *16th international ieee conference on intelligent transportation systems (itsc 2013)* (pp. 1456–1461).

Downsland, K. A., & Greaves, A. M. (1994). Collision avoidance in bi-directional agv systems. *Journal of the Operational Research Society*, *45*(7), 817–826.

Dresner, K., & Stone, P. (2008). A multiagent approach to autonomous intersection management. *Journal of artificial intelligence research*, *31*, 591–656.

Fanti, M. P. (2002). Event-based controller to avoid deadlock and collisions in zone-control agvs. *International journal of production research*, *40*(6), 1453–1478.

Grunow, M., Günther, H.-O., & Lehmann, M. (2006). Strategies for dispatching agvs at automated seaport container terminals. *OR spectrum*, *28*(4), 587–610.

Ho, Y.-C., & Liao, T.-W. (2009). Zone design and control for vehicle collision prevention and load balancing in a zone control agv system. *Computers & Industrial Engineering*, *56*(1), 417–432.

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

Liu, C.-I., Jula, H., Vukadinovic, K., & Ioannou, P. (2004). Automated guided vehicle system for two container yard layouts. *Transportation Research Part C: Emerging Technologies*, *12*(5), 349–368.

Manca, S., Fagiolini, A., & Pallottino, L. (2011). Decentralized coordination system for multiple agvs in a structured environment. *IFAC Proceedings Volumes*, *44*(1), 6005–6010.

Möhring, R. H., Köhler, E., Gawrilow, E., & Stenzel, B. (2005). Conflict-free real-time agv routing. In *Operations research proceedings 2004* (pp. 18–24).

Qiu, L., Hsu, W.-J., Huang, S.-Y., & Wang, H. (2002). Scheduling and routing algorithms for agvs: a survey. *International Journal of Production Research*, *40*(3), 745–760.

Ter Mors, A. W. (2009). *The world according to marp* (Unpublished doctoral dissertation). Delft University of Technology.

Viswanadham, N., Narahari, Y., & Johnson, T. L. (1990). Deadlock prevention and deadlock avoidance in flexible manufacturing systems using petri net models. *IEEE transactions on robotics and automation*, *6*(6), 713–723.

Wu, N., & Zhou, M. (2001). Resource-oriented petri nets in deadlock avoidance of agv systems. In *Proceedings 2001 icra. ieee international conference on robotics and automation (cat. no. 01ch37164)* (Vol. 1, pp. 64–69).

Yoo, J.-w., Sim, E.-S., Cao, C., & Park, J.-W. (2005). An algorithm for deadlock avoidance in an agv system. *The International Journal of Advanced Manufacturing Technology*, *26*(5-6), 659–668.

# 8 Appendix

## 8.1 Overview of notation used

### 8.1.1 Sets

$V$ : set of vertices on the terminal

$B \subset V$ : set of buffer locations on the terminal

$A$ : set of arcs on the terminal

$H$ : set of roads or lanes on the terminal

$S(a)$ : set of arcs blocked when the center of an AGV is on arc $a \in A$

$P$ : set of requests

$K$ : set of AGV's

$R$ : set of routes

$J$ : set of loads

### 8.1.2 Variables and others

$G = (V, A)$ : graph representing the terminal with vertices and arcs

$d_{ij}$ : distance between vertices $i \in V$ and $j \in V$, with $(i, j) \in A$ (meters)

$v_{ij}^{max}$ : maximum allowed speed on arc $(i, j) \in A$ (m/s)

$a_k$ : maximum acceleration of AGV $k \in K$ (m/s$^2$)

$b_k$ : maximum deceleration of AGV $k \in K$ (m/s$^2$)

$o_p$ : origin of request $p \in P$

$d_p$ : destination of request $p \in P$

$\theta_p$ : start time of request $p \in P$

$a_{rp}$ : 1 if route $r \in R$ feasible contains request $p \in P$

$b_{rijt}$ : 1 if route $r \in R$ uses arc $(i, j) \in A$ at time $t$

$c_r$ : completion time of route $r \in R$ (seconds)

$x_r$ : 1 if route $r \in R$ is used

$G' = (V', A')$ : graph in which vertices also include the speed with which a vehicle arrives at the node.

$z_{kt}$ : coordinate of AGV $k \in K$ at time $t \in T$

$x$ : center x-coordinate of an AGV

$y$ : center y-coordinate of an AGV

$\gamma_r$ : rotation coordinate of an AGV (radians)

$x_0$ : x-coordinate of the center front of an AGV

$y_0$ : y-coordinate of the center front of an AGV

$x_{rf}$ : x-coordinate of the right front of an AGV

$y_{rf}$ : y-coordinate of the right front of an AGV

$x_{lf}$ : x-coordinate of the left front of an AGV

$y_{lf}$ : y-coordinate of the left front of an AGV

$x_{rr}$ : x-coordinate of the right rear of an AGV

$y_{rr}$ : y-coordinate of the right rear of an AGV

$x_{lr}$ : x-coordinate of the left rear of an AGV

$y_{lr}$ : y-coordinate of the left rear of an AGV

$D$ allowed distance or trajectory length (meters)

$v_0$ : velocity and the start of the trajectory (m/s)

$v_1$ : desired ending velocity (m/s)

$v_j^{max}$ : maximum velocity given the trajectory and load $j \in J$ (m/s)

$v^*$ : highest velocity reached on the trajectory (m/s)

$D^*$ : distance travelled at constant speed on the trajectory (meters)

$t^*$ : time needed to traverse the trajectory of length D (seconds)

$v_g, v_r$ : current speed of AGV's $g \in K$ and $r \in K$ $(m/s)$

$a_g, a_r$ : maximum acceleration of AGV's $g \in K$ and $r \in K$ with their current load $(m/s^2)$

$b_g, b_r$ : maximum deceleration of AGV's $g \in K$ and $r \in K$ with their current load $(m/s^2)$

$d_{gr}$ : Euclidean distance between vehicles $g \in K$ and $r \in K$ (meters)

$t_{gr}$ : time it takes vehicle $g \in K$ to travel $d_{gr}$ at its current speed (seconds)

$d_r(t_{gr})$ : distance traveled by $r \in K$ during $t_{gr}$ when braking at full force from its current speed (meters)

$D_g$ : allowed distance of vehicle $g \in K$ (meters)

$M$ : distance margin between two AGV's (meters)

$t_{delay}$ : communication time between AGV's (seconds)

$t^-$ : claim margin with two way communication

$c_{block}$ : first claim a horizontal AGV encounters for which it can stop in time, but that it cannot cross in time

$c_{stop}$ : the latest claim before which an horizontal AGV can stand still without occupying a claimed area

## 8.2 Abbreviations

QC: Quay Crane

RTG: Rubber Tired Gantry

TEU: Twenty foot Equivalent Unit

ECT: Europe Container Terminal

AGV: Automated Guided Vehicle

TBA: Technisch Bestuurskundige Adviesgroep (Technical and Managerial Consuntancy Group)

VRP: Vehicle Routing Problem

MARP: Multiple Agent Routing Problem

SPPTW: Shortest Path Problem with Time Windows

RMG: Rail Mounted Gantry

TP: Transfer Point

ASC: Autonomated Stacking Crane