Design and analysis of a platooning strategy for dense grid-based AGV systems

van Willigen, J.M.

Award date:
2020

Design and analysis of a platooning strategy for dense grid-based AGV systems

Department of Mechanical Engineering
MSc Manufacturing Systems Engineering
Control Systems Technology

Author:
J.M. van Willigen
0889156

Supervisors:
ir. G.J.A.M. Weijenberg
dr.ir. J.A.W.M. van Eekelen

February 3, 2020

Preface

This thesis concludes my graduation project for the master Mechanical Engineering at Eindhoven University of Technology. In my years as a student, I further developed my technical knowledge, my social and communicative skills, and more importantly, I met a lot of new interesting people, many of whom I now call friends.

For nine months, I have worked on my graduation project at Vanderlande. I am very fortunate to have been able to add to an interesting and challenging project. In my time at Vanderlande, I got to know Vanderlande as an open, friendly, and stimulating working environment.

I am especially thankful for the support of my thesis supervisors Gaston Weijenberg, and Joost van Eekelen. Gaston has taught me a lot in how to more clearly and intuitively communicate my ideas with others. From Joost I learned the important of structure and details when working on a large project with multiple people. Additionally, both Gaston's and Joost's critical feedback stimulated me to go that extra mile. Also, I would like to thank my family and friends for their unconditional support throughout my entire study period, in particular my parents, brother, friends from TU/e, my home town Boxmeer, music association Quadrivium, and my (former) house mates.

Abstract

Vanderlande is currently designing an AGV system, multi-applicable to the airports, parcel, and warehousing industry. The goal of the system is to create a flexible, robust, and scalable alternative to systems that consist of fixed-world equipment such as conveyor belts, which are currently the standard in these industries. The AGV system is a grid-based system, meaning AGVs in the system can only travel in the directions defined by a grid to deliver items from A to B. A grid consists of tiles, with segments defining the allowed driving directions for the AGVs. To meet throughput demands, high AGV densities, 20% or higher are required, i.e. per 100 tiles 20 or more AGVs are in the system. Platooning is a concept invented for road traffic to increase traffic efficiency. The concept is anticipated to also benefit dense grid-based AGV systems. Therefore, to further increase system performance this thesis proposes a grid-based platooning strategy, and analyzes its effects on the system.

To the best of knowledge, no platoon control strategies exist applicable to grid-based systems. Therefore, this thesis proposes several platooning strategies for grid-based systems. A strategy is designed for AGVs driving in a straight lane, as well as for AGVs cutting a corner: straight-line platooning and curved-line platooning, respectively. Both strategies are based on the leader-follower principle. In this principle, a leader vehicle can indicate for one or multiple of its reserved tiles, if these tiles will become free in the near future by giving or not giving an indication for this tile. A follower vehicle, considering a tile for which an indication is given, can follow-up on this indication by taking over the tile reservation under certain conditions. These conditions ensure safety. In a grid-based system, a turn can be made by pivoting at standstill at the center of a tile, or by cutting a corner. Curved-line platooning is only possible if the leader vehicle decides to turn by cutting the corner. Therefore, a strategy is designed that allows AGVs to cut a corner as often as possible: lenient cutting corners. Straight-line and curved-line platooning share the negative side effect that AGVs outside of a platoon can have to wait for a long platoon to pass, before being allowed to continue. To mitigate this effect, three coordination strategies are designed.

The performance of all the designed grid-based platooning strategies is investigated using scenario analyses. Additionally, straight-line platooning and the three coordination rules are implemented in a MATLAB simulation model to allow for simulation analysis. This thesis distinguishes grid-type layouts and segment-type layouts. A grid-type layout is a layout where the majority of the tiles has multiple incoming segments and multiple outgoing segments; a segment-type layout is a layout where the majority of the tiles has

one incoming segment and one outgoing segment. Also rectangular layouts and hexagonal layouts[1] are distinguished, respectively consisting of rectangular and hexagonal tiles. The scenario and simulation analyses suggest substantial positive effects can be achieved for square grid-type layouts, and for square segment-type layouts by grid-based platooning. Hexagonal layouts, however, show only a small, but significant benefit from grid-based platooning.

[1]For the hexagonal grid layouts a patent is pending

Contents

1: Introduction

The production, trade and service industry are undergoing significant change as a result of a paradigm shift towards flexible automation. Flexible automation is the ability for an automated system to quickly and easily evolve or be re-tasked [1]. Vanderlande is a company accommodating this shift in industry, being a globally leading company in the field of value-added logistic process automation. Vanderlande operates actively in three market segments: airports, parcel, and warehousing.

1. Within the airports industry, passenger numbers and aircraft movements are growing. Therefore, airports need to invest in optimizing their internal logistic processes.
2. The parcel industry's service level demands are ever-increasing. Parcel handling companies, such as DHL and FedEx are therefore investing in further automating and optimizing their processes, such as screening and sortation of parcels.
3. Performance demands are also growing for the warehousing industry. Especially as a result of e-commerce, companies are investing in flexible automation.

All three market segments share the need for efficient material handling systems, to transport items between processes. Traditionally, automated handling systems with 'fixed-world' equipments, such as roller or chain conveyors, are often used in value-added logistics. Although these type of systems have proven to perform well in system throughput, in the near future, they might not be able to provide the flexibility, scalability, and robustness required by industry, e.g. installing a conveyor-belt system requires a significantly long setting-up time, and making system alterations is difficult. Additionally, the effect of a conveyor belt breaking down is largely felt, as the whole system must shut down until the blockage is resolved.

An alternative for fixed-world automated handling systems uses autonomous mobile robots to move items from point A to B. These robots show potential due to their flexibility regarding integration in existing and changing environments; paths can be redefined without the infrastructure changes required by conveyor systems, thus offering easier ways of adapting an existing system. Such a robot is called an Automated Guided Vehicle (AGV), see Figure 1.1. These AGVs are, in accordance with the earlier mentioned paradigm shift, increasingly often used in industrial environments such as manufacturing facilities and warehouses [2]. Already advanced example systems include Kiva systems by Amazon Robotics [3], FLEET by Vanderlande [4], and the AutoStore system [5]. In China a grid-based AGV system handles package sortation in an Alibaba warehouse [6].

Figure 1.1: Example AGV used to transport baggage at airports. [4]

1.1 *AgvSorter* system

Because of the potential shown by mobile infrastructure, Vanderlande is currently designing an AGV system, called the *AgvSorter*. No physical version or prototype of the system has been developed yet. However, Vanderlande has created a simulation model in MATLAB. This model is applicable to all three segments Vanderlande is active in: airports, parcel, and warehousing. The *AgvSorter* is a grid-based system, meaning that AGVs in this system can only travel in the directions that are defined by a predefined grid. This grid consists of tiles, with segments defining the allowed driving directions for the AGVs. These tiles are 'virtual', meaning they do not have any physical purpose. An AGV can only travel over a tile, if it has claimed this tile. Each AGV executes jobs. A job consists of going to a source, to pick up an item, and then going to a destination, to drop off this item.

In the further development of the *AgvSorter*, using the MATLAB simulation model, model-based design techniques can be applied. These techniques are powerful tools in the development of cyber-physical systems [7], and allow to validate, verify, and test system components that are not yet physically realized. Using model-based design techniques, a system can be optimized on performance. Performance for the *AgvSorter* is influenced by several factors, such as system layout, grid-size, number of AGVs, and operational transportation control. Operational transportation control consists of task allocation, path planning, and traffic control:

1. Task allocation assigns tasks to AGVs, i.e. source and destination allocation and battery recharge procedures.

2. Path planning includes determining the path of the AGV necessary to transport a job from source to destination.

3. Traffic control ensures AGVs to drive conflict-free, while following their path. This includes assuring no collisions, and no deadlocks occur. In case of grid-based AGV systems, this means deciding whether or not an AGV is allowed to reserve the next tile on its path.

Using the simulation model, the effects of new control strategies and system configuration on the performance can be investigated. This thesis focuses on traffic control. Traffic control gets increasingly more difficult for denser systems. In order to achieve throughput demands set by industry, high AGV densities are necessary. Densities of 20% or higher are considered high, meaning per 100 tiles 20 or more AGVs are in the system. Hence, this thesis focuses on systems with densities of 20% and 25%.

1.2 Road traffic control and platooning

There are considerable parallels between road traffic control and traffic control of a grid-based AGV system, hence studies towards road traffic optimization might prove useful for grid-based AGV systems as well. A notable similarity between road traffic and grid-based AGV systems is the tendency of congestions to form where two or more traffic flows merge, or where traffic flow needs to slow down, e.g. intersections and corners. For road traffic control, studies have shown that platooning can significantly reduce travel times, and mitigate congestions [8][9][10]. Therefore, it is deemed worthwhile to investigate if the concept of platooning can achieve a similar result for grid-based AGV systems, and increase system performance. However, to the best of found knowledge, no research has been performed towards this goal yet.

1.3 Objective

This thesis focuses on designing a platooning strategy for the *AgvSorter* model, and investigates the effects of this strategy on the system's performance. The objective of this thesis is:

> Design a platooning strategy for the dense *AgvSorter* system, and analyze its effects on system performance.

Until now in this thesis, performance has not been properly defined. However, in order to evaluate different system control designs, this is essential. To this extent, Key Performance Indicators (KPIs) are defined for the *AgvSorter* system.

- *System throughput:* the main KPI of the *AgvSorter* is the system throughput, which is defined as the number of items delivered at the correct destination per time unit.
- *Bottleneck throughput:* if we zoom in at pieces of a grid-based AGV system, we can identify several scenarios. The bottleneck throughputs of these scenarios are relevant indicators for the system performance. Bottleneck throughput is defined as the throughput of a scenario, if it were to be a bottleneck in a larger system. This means an AGV always wants to enter the scenario, and can always leave the scenario.

- *Lead time percentile:* the lead time percentile is defined as the 95% percentile of job lead times. It is undesirable for any item to remain in the system for too long; a job lead time distribution with a small tail above the mean is desired. Therefore, a relevant KPI is the lead time percentile.

The following four quality attributes are especially important for the *AgvSorter*:

- *Safety:* the *AgvSorter* must prevent the occurrence of collisions between AGVs, and between AGVs and external objects.
- *Scalability:* the ability of the system to handle bigger workloads by adding resources to the system, e.g. AGVs, loading stations, drivable area.
- *Flexibility:* the ability to handle changes in system configuration, such as a different layout.
- *Robustness:* the ability of the system to handle with local system malfunction.

The goal of designing a grid-based platooning strategy is to improve KPIs, without compromising on quality attributes.

1.4 Outline

The remaining six chapters of this thesis work towards achieving the thesis' objective. The contents of these chapters are briefly discussed below.

Chapter 2: Platooning - To be able to design a grid-based platooning strategy, research is conducted towards existing platooning systems. The results of this research are set out in Chapter 2.

Chapter 3: *AgvSorter* system - In this chapter, a detailed description of the *AgvSorter* system is given, focusing on the control model and the simulation model. The chapter also introduces benchmark scenarios and system configuration with which different versions of the *AgvSorter* system can be compared. The chapter ends with discussing the potential of platooning for grid-based AGV control.

Chapter 4: Grid-based platooning - The design of several platooning strategies are detailed in this Chapter. For every strategy the desired behavior, the strategy design, and the resulting behavior are discussed.

Chapter 5: Implementation - Several of the discussed platooning strategies are implemented in the *AgvSorter* MATLAB simulation model. This chapter elaborates on the verification and validation steps that are performed to ensure that the implemented strategies are correctly implemented, and that the simulation model gives relevant results.

Chapter 6: Results & discussion - In this chapter the different platooning strategies are compared against the baseline *AgvSorter* model, using the benchmark scenarios and the benchmark system configurations.

Chapter 7: Conclusions & recommendations - Conclusions are drawn regarding the performance of the different platooning strategies. Recommendations are made towards further research into grid-based platooning and application of the designed grid-based platooning strategies.

2: Platooning

Platooning is a method invented for road traffic to increase traffic and fuel efficiency. Especially car and truck platooning is of broad and current interest to researchers [11]. Consequently, in literature, platooning is often defined in the context of traffic involving trucks and cars. The concept of platooning might also be applied in a grid-based AGV system. This project focuses on grid-based AGV systems. Hence, the following definition that fits this context is selected from literature. This definition is used for the remainder of this thesis.

Definition: *A platoon is a string of automated vehicles, which drive in a close spacing behind each other without violating safety requirements.* [12]

In this definition a string of vehicles refers to a group of vehicles driving in a row, with each vehicle following the acceleration and deceleration behavior of its predecessor. In [12], platoon research has been subdivided into two categories: platoon control and platoon coordination. Platoon control deals with intra platoon interaction. This includes lateral and longitudinal control. Platoon coordination deals with inter platoon interaction. This encompasses platoons interacting with other platoons or vehicles. The remainder of this chapter first discusses platoon control, and then discusses platoon coordination.

2.1 Platoon control

Platoon control deals with intra platoon interaction. It can be subdivided into longitudinal control and lateral control [13]. Longitudinal controllers are responsible for regulating vehicle velocity. Lateral controllers take care of steering the vehicle for path tracking. This thesis focuses on longitudinal control, as the AGVs follow a grid. The main tasks of longitudinal controllers are to ensure collision-free driving, minimize inter-vehicular distance, and guarantee string stability. String stability is defined in [14] as follows:

Definition: *The property string stability ensures that any perturbation of the velocity or position of the leading vehicle will not result in amplified fluctuations to the following vehicle's velocity and position.*

For longitudinal controllers to be able to achieve their main tasks, state information from other vehicles in the platoon is necessary. Therefore, vehicle-to-vehicle communication is essential. According to [15], only when the position, speed, and acceleration of the leading

vehicle is used as reference information, one can achieve string stability. But having just this information about the leading vehicle is not sufficient to ensure collision-free driving. For this the position, speed, and acceleration of the immediately preceding vehicle are necessary. This communication strategy is shown in Figure 2.1. [15]

Figure 2.1: Vehicle-to-vehicle communication strategy to achieve collision-free driving and string stability [15]

Recent research focuses on designing longitudinal control strategies to further minimize inter-vehicular distance, ensure collision-free driving, and guarantee string stability. CACC uses a leader-follower approach, meaning the leader vehicle is used as a reference for the follower vehicles. This approach depends on the reliability of the leaders, and does not allow feedback from the follower to the leader [16]. An alternative approach, proposed in [17], is based on a reference point that moves according to the mission, and functions as a virtual leader for the platoon. In [18], Shida and Nemoto proposed a strategy in which the platoon is modeled as one dynamic system, and applied optimal linear quadratic control theory to the problem. String stability, excellent following performance and smooth driving were confirmed.

To the best of knowledge, no longitudinal controller applicable to grid-based control has been described in literature. An advantage of a grid-based platooning strategy, compared to strategies found in literature, is that it does not require vehicle-to-vehicle communication, but only communication with a central controller.

2.2 Platoon coordination

Platoon coordination tries to regulate inter platoon interaction. This includes platoons interacting with other platoons or vehicles. Therefore, platoons need to be able to communicate with other platoons and/or central controllers. This communication is called vehicle-to-infrastructure communication. Platoon coordination can be decentralized, meaning local control; centralized, meaning central control; or a hybrid version of the two. Centralization makes it possible to globally optimize. Decentralized coordination is often chosen, because it is less complex as it involves the control of less entities. Platoon coordination involves coordinating platoons as efficiently as possible through the system. Important tasks of platoon coordination are platoon manoeuvres and coordination of platoons through bottlenecks.

Manoeuvres

Platoon manoeuvres are actions that alternate the vehicle composition of a platoon. A list of platoon manoeuvres is given:

- Form: two vehicles form a platoon
- Join: a vehicle joins a platoon
- Merge: two platoons merge into one platoon
- Leave: a vehicle leaves a platoon
- Split: one platoon splits into two platoons
- Break up: a platoon breaks up (no platoon remains)

The first three manoeuvres are constructive, and the last three manoeuvres are destructive manoeuvres. Different strategies on when to perform a constructive manoeuvre exist. A simple strategy is to adopt the speed of a preceding vehicle [19]. In [20], different strategies on when to perform a constructive manoeuvre are developed and analyzed. Their work focused on maximizing the time before a destructive manoeuvre. The described strategies include clustering of vehicles based on their destination, and ordering vehicles based on how long they can travel along with the platoon. Manoeuvres often occur at bottlenecks, as vehicles accumulate, meet, and diverge here. A way of increasing performance is to actively form platoons before reaching a bottleneck. This platoon forming is achieved by vehicles adjusting their speed. As multiple platoons approach a bottleneck, a central controller tries to schedule the arrival of the platoons, so neither one of them has to fully stop [8]. Hoshino [21] improved this method by basing platoon formation on a reactive clustering method, and showed the effectiveness of his proposed strategy.

Bottlenecks

Merge junctions and intersections are the principal bottlenecks in vehicle systems [22]. Therefore, numerous researchers have investigated and analysed the effect of platoons on throughput at these bottlenecks.

- *Intersections*: Jin et al. [8] showed applying their platooning strategy, following a leader-follower approach, at intersections improved travel time by 30%. Miculescu et al. [23] considered an intersection as a polling system, with one server and two queues. They showed using simulations, that in a heavy load system, platooning behavior emerged as the optimal solution. At intersections, one can think of different priority rules regarding platoons. One that seems logical is: 'Give priority to the biggest platoon.' However, if we are trying to minimize maximal delay, it is not always optimal to allow a platoon to cross, even when there is no platoon on the other incoming lane that can cross at the same time. An example is given in Figure 2.2. If the long platoon crosses first, the short platoon is delayed relatively long. To minimize maximal delay, it might be better to give priority to the small platoon. However, following the second strategy, the time until both platoons have cleared the crossing will be longer, potentially causing greater delays for later platoons. In [9] it is suggested that, to determine an appropriate strategy, if we know the maximum delay that we are willing to tolerate, we can apply a simple greedy algorithm minimizing the time until the crossing is cleared. [9]

- *Merge junctions*: Another interesting strategy, called virtual platooning, is proposed in both [24] and [25]. In this strategy, when two vehicles are approaching a merge, the vehicle from one lane is mapped on the other lane. If the virtual vehicle keeps a safe distance from the 'real' vehicle, the vehicles can merge safely. In [22] it is also shown a platooning strategy at merge intersections can increase vehicle throughput, and additionally investigates the influence of platoon size. In this study, a vehicle string of 10 can provide 5.7 times the lane capacity of a single vehicle string. Diminishing improvements per added vehicle set in as string lengths increase, so that it may not be advantageous to increase the string lengths indefinitely.

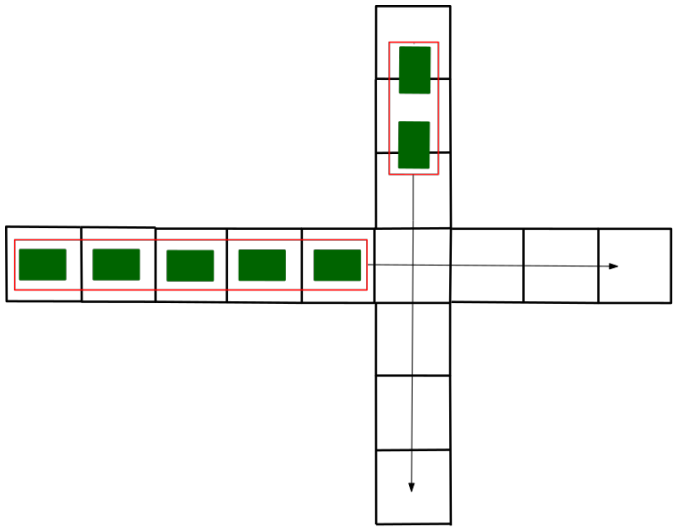


Figure 2.2: An example where crossing if possible is not always the best choice; the red rectangles represent the platoons

2.3 Summary

Platoon research can be subdivided into two categories: platoon control and platoon coordination. In platoon control, important tasks are ensuring collision-free driving, minimizing inter-vehicle distance, and guaranteeing string stability. This is achieved using longitudinal control. Various longitudinal controllers, with various performance, exist in literature. However, to the best of knowledge, no longitudinal controller applicable to grid-based control, or anything similar, has been described in literature. In platoon coordination important tasks include regulating platoon manoeuvres and regulating platoons and vehicles efficiently through bottlenecks in the system. For a grid-based platooning strategy, such platoon coordination strategies might improve system performance, and can be used as inspiration in the design process.

3: *AgvSorter* system

Before designing a grid-based platooning strategy, in this chapter, the baseline *AgvSorter* is analyzed. First, the control model and the simulation model are described. A more elaborate description is given for traffic control. Next, ten benchmark scenarios and three benchmark system configurations are introduced. Lastly, the potential of platooning for grid-based AGV control is shown for the *AgvSorter* system.

3.1 Control model

As before mentioned, the *AgvSorter* model is controlled using grid-based control. This means that AGVs in the system can only travel in the directions that are defined by the grid, consisting of tiles and segments. The model allows two types of tile geometries: rectangular and hexagonal[2]. The geometry and configuration of the tiles, and how they are connected with segments are system design choices, e.g. Figure 3.1 and 3.2. All segments are allocated a maximum allowed AGV speed: the segment speed.

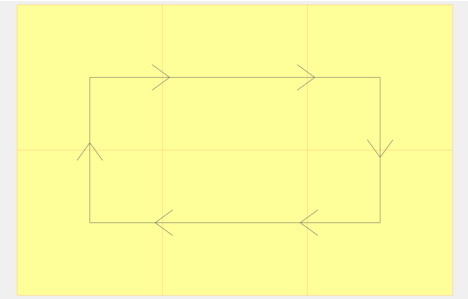


Figure 3.1: A simple configuration with rectangular tiles

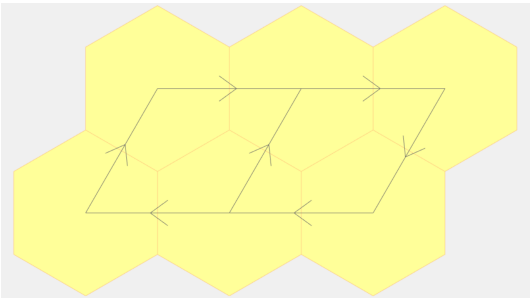


Figure 3.2: A simple configuration with hexagonal tiles

Grid-based control is based on seven ground principles, or rules:

1. AGVs move from tile to tile.
2. Tiles do not overlap geometrically.
3. AGVs can pivot within the borders of the tile.
4. Each tile can be reserved by one AGV at a time.
5. An AGV is only allowed to move over tiles that have been reserved for that AGV.

[2]For the hexagonal grid layouts a patent is pending

6. When moving from tile A to tile B, the AGV shall not cross another tile.
7. An AGV is only allowed to move over a tile in correspondence to the direction specified by the segment(s) corresponding to that tile.

In addition to these ground rules, this project also builds upon certain assumptions.

1. The uni-directional segments assumption: a segment can not allow AGVs to move in two directions over the segment.
2. The perfect world assumption: An AGV never malfunctions; there is no communication loss between different entities in the system; the motion control of the AGVs is perfect, etc.
3. The identical AGVs assumption: all AGVs are identical.

Each AGV can execute a job, for which it will first go to a source, to pick something up, and then go to a destination, to drop something off. In Chapter 1, the three main tasks of operational transportation control were introduced: task allocation, path planning, and traffic control. This project focuses on traffic control, hence, later in this chapter, a separate section is dedicated to traffic control. To get a better understanding of the model, task allocation, and path planning are now briefly discussed.

- *Task allocation:* the process of allocating a task to an AGV starts with an AGV becoming idle. The AGV is then assigned the task associated with the closest source with an available task. Only a maximum, predefined number of AGVs can be assigned a task at the same source.
- *Path planning:* the *AgvSorter* model makes use of a path planning approach that avoids congestions. The approach uses a graph-representation with vertex and edge weights, representing the time necessary to respectively pass tiles and segments. These weights are updated over time. The graph-representation allows to determine a shortest-time path, using an adapted version of the $A*$-algorithm, that also takes turning time into account. This shortest-time path is dynamically updated to deal with new and dissolved congestions, see [26].

3.2 Simulation model

To gain insight in the workings and performance of the *AgvSorter*, a simulation model has been developed in MATLAB by Vanderlande. The simulation model is a discrete-event simulator, meaning the model behavior is simulated at discrete points in time at which an event is scheduled for a simulated object, such as AGVs or the controller. Example events include an AGV leaving a tile, or an AGV finishing loading.

In the simulation model of the *AgvSorter*, a valid system layout requires sources and destinations. In Figure 3.3, an example visualization of such an *AgvSorter* system layout is shown, containing the basic elements of the model.

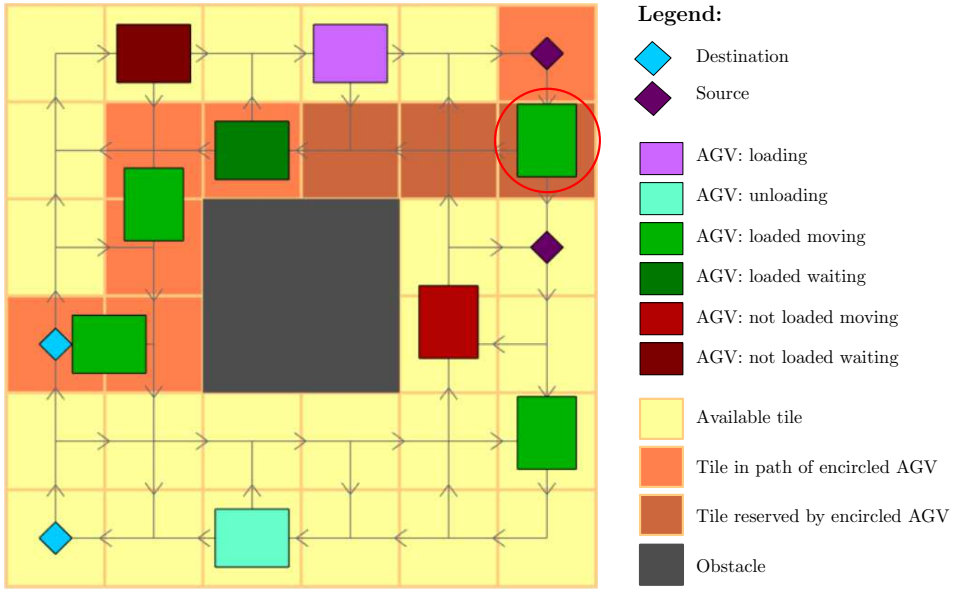


Figure 3.3: Example visualization of a grid-layout of the *AgvSorter* system

The shown layout consists of a square grid with 36 tiles, nine AGVs, three destinations, and three sources. AGVs are represented using rectangles, in various colors depending on the state of the AGV. Possible states are 'loading', 'unloading', 'loaded moving', 'loaded waiting', 'not loaded moving', and 'not loaded waiting'. Sources and destinations are represented by blue and purple colored diamonds, respectively. However, not all destinations and sources are visible, due to AGVs on top of them. The grey square in the middle of the layout represents an obstacle. In Figure 3.3, the encircled AGV is selected, and as a result its path is indicated by orange, and its reserved tiles are shaded with a darker orange.

3.3 Traffic control

A major challenge in a dense AGV system is traffic control. A traffic controller must guarantee collision-free and deadlock-free driving. The baseline *AgvSorter* allows a tile to be reserved by one AGV at a time. An AGV is only allowed to move over tiles that have been reserved for that AGV. This is achieved by only allowing AGVs to drive over a segment if the tile this segment leads to is reserved by the AGV. In the baseline *AgvSorter* model, there are two mechanisms that instigate tile reservation: (1) an AGV asks for tiles to reserve; (2) when a tile is released by an AGV, from all AGVs waiting for this tile, an AGV is selected that may reserve the tile.

Instigation mechanisms for tile reservation

Mechanism 1: an AGV asking for tiles to claim can be triggered by several events: (1) the AGV finished (un)loading; (2) the AGV finished traversing a segment; (3) the AGV finished turning. (4) the AGV reroutes. If one of these events occurs the AGV tries to claim a specified number of tiles (*nPosToClaim*) ahead.

Mechanism 2: if an AGV releases a tile, the tile is generally free to be reserved by other AGVs. Sometimes multiple AGVs are waiting for the same tile. In this case it is necessary to select which AGV may reserve the tile. If the tile becomes free, the flow chart shown in Figure 3.4 is commenced.

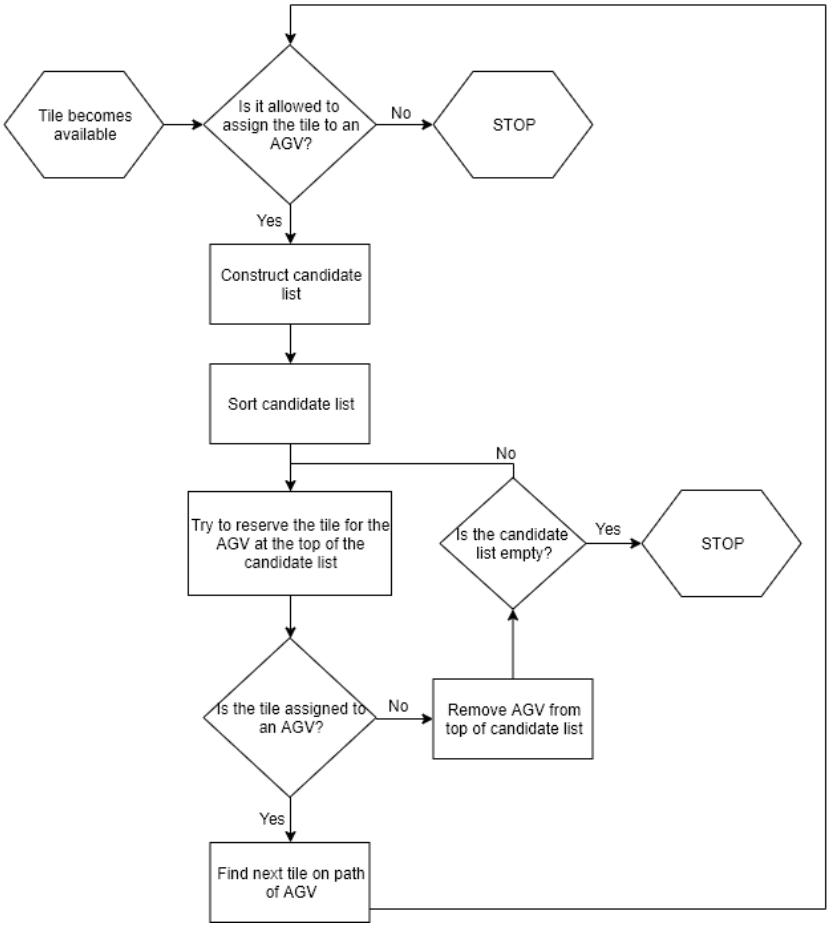


Figure 3.4: A flow chart of what happens when a tile becomes available

First, it is checked if it is allowed to assign the tile to an AGV, i.e. is the tile unclaimed. Subsequently, a candidate list, consisting of AGVs waiting for this tile, is constructed. An AGV is considered to be waiting on a tile if the tile is within *nPosToClaim* tiles reach, and it is the first tile to claim after the already claimed tiles. The candidate list is then sorted based on simple decision rules. A list of these decision rules is given in order of priority:

1. The AGV carrying the job with the highest priority goes first.
2. The AGV closest to the point of conflict goes first.
3. A moving AGV goes first.
4. The AGV that is waiting longest goes first.
5. A random tie breaker is used to determine order.

The AGV at the top of the candidate list is selected, and for this AGV it is tried to make a reservation. However, in order to prevent deadlocks, it may occur that the AGV is not allowed to reserve the tile. Then, the AGV is removed from the candidate list, and the process is re-instigated for the next AGV at the top of the candidate list, until a reservation for the tile is made, or the candidate list is empty. If the tile is reserved, the next tile of the claimer is determined. For this tile the same procedure is again followed. If the claimer has already reserved *nPosClaim* tiles ahead, or has no next tile on its path, the flow charts stops.

Tile reservation

Both in instigation mechanism 1 and instigation mechanism 2, AGVs try to reserve tiles. If an AGV tries to make a reservation, it is not always allowed to do so, e.g. a tile is already claimed by another AGV, or claiming the tile would cause a deadlock situation. In Figure 3.5 the flow chart is shown of the process of tile reservation for an AGV.

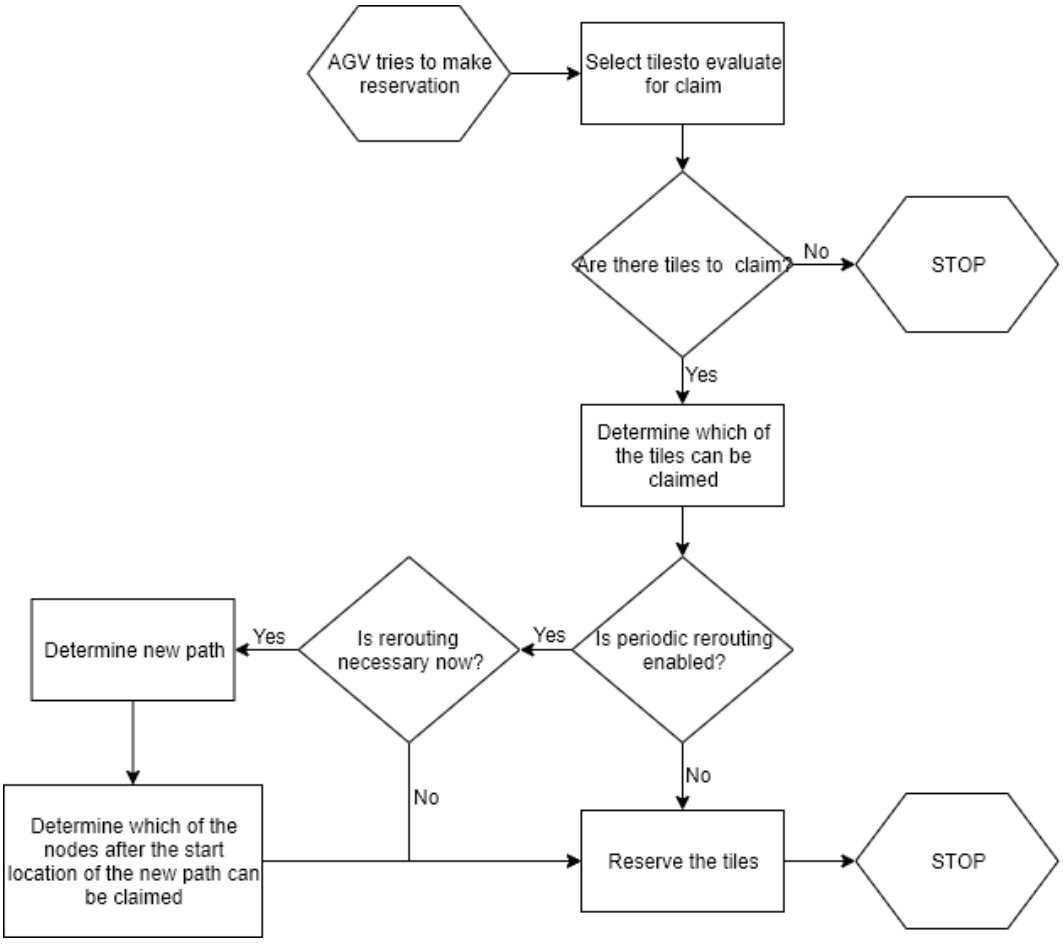


Figure 3.5: The flow chart of what happens when an AGV tries to make a reservation

First, the tiles the AGV wants to reserve are selected. If at least one tile is selected, the next step is to determine which tiles can be claimed. These tiles are determined by executing four consecutive checks:

1. Determine which of the tiles evaluated have not been claimed yet, and select the tiles until the first claimed tile.
2. Determine which of the selected tiles need to be claimed in a group. If not all tiles in a group can be claimed, then none of the tiles should be claimed. Grouped claims can be necessary for on-the-fly (un)loading. On-the-fly (un)loading is a type of (un)loading where the AGV is loaded from above, and does not need to come to a complete standstill.
3. Determine which of the remaining tiles can be claimed without causing a deadlock, and select those. How is checked for deadlocks will be elaborated on later in this chapter.
4. Determine which of the selected tiles need to be claimed in a group. If not all tiles in a group can be claimed, then none of the tiles should be claimed. This step is identical to step two. However, as step three might change the selected tiles, it must be performed again.

The *AgvSorter* model has the optional feature to periodically replan the AGVs' paths. If periodic rerouting is enabled, and rerouting is necessary, a new path is determined, and, using the previously mentioned four steps, it is checked if tiles on the new path can be reserved. The next step is to actually reserve the selected tiles, thus finalizing the flow-chart.

Resulting local behavior

It is interesting to analyze the local behavior that is the result of the baseline traffic control. In Figure 3.6 a congested crossing is shown. The priority rules and their order enforce that alternately from different directions one AGV is allowed on the tile. This is visualized in the figure by means of numbers (lowest number goes first). AGVs from different directions alternately crossing a tile, is a phenomenon seen in all scenarios where multiple traffic flows merge. Applying platooning would result in a more efficient local behavior, where AGVs can cross a crossing in groups with small inter-vehicular distances.

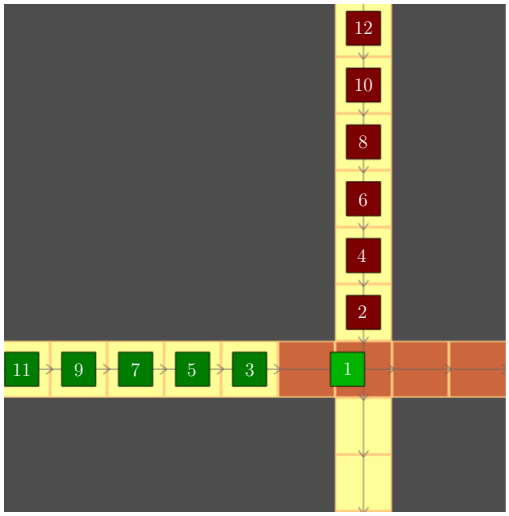


Figure 3.6: A visualization of the local behavior at a congested crossing.

Cutting corners

Under normal circumstances, when an AGV wants to make a turn, the AGV fully stops to rotate in the center of the corner tile. This strategy is called pivoting, but it is time consuming. Therefore, the baseline *AgvSorter* model provides an optional feature called 'cutting corners'. A turn is made by following a curve without coming to a complete stop. Three types of corners are recognized: a single turn, an S-curve, a U-turn. AGVs recognize these turns by looking at their reserved tiles. In Figure 3.7, the three possible opportunities for corner cutting are shown. A corner is only cut if the AGV has claimed all tiles matching the turn. Therefore, when an AGV is driving closely behind another AGV, it will not cut the corner. A platooning strategy that allows AGVs to cut a corner as a platoon can decrease the occurrence of this phenomenon, thus positively affecting system performance.

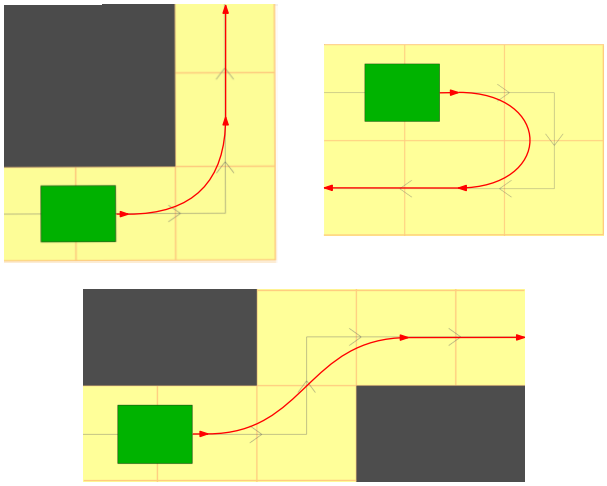


Figure 3.7: The three opportunities for cutting corners for a single turn (top-left), an S-curve (bottom-middle), and a U-turn (top-right)

3.4 Benchmark scenarios

Grid-based AGV systems quickly become complex. However, such a system comprises certain basic scenarios, such as crossings, corners, and loading points. These basic scenarios are identified, and translated into benchmark scenarios. The bottleneck throughput is determined for all these benchmark scenarios, for both pivoting and cutting corners. The bottleneck throughput is defined as the throughput of a scenario, if that scenario were to be a bottleneck of a large system. This means the next AGV is always waiting to enter the scenario, push behavior, and an AGV is always able to leave the scenario, pull behavior. It is chosen to look at the throughput of incoming lanes, as this is where AGVs accumulate. A list of these benchmark scenarios is given in Appendix A. This appendix shows illustrations, along with corresponding configuration parameters of the scenarios. In Figure 3.8 the bottleneck throughputs are visualized in a bar plot.

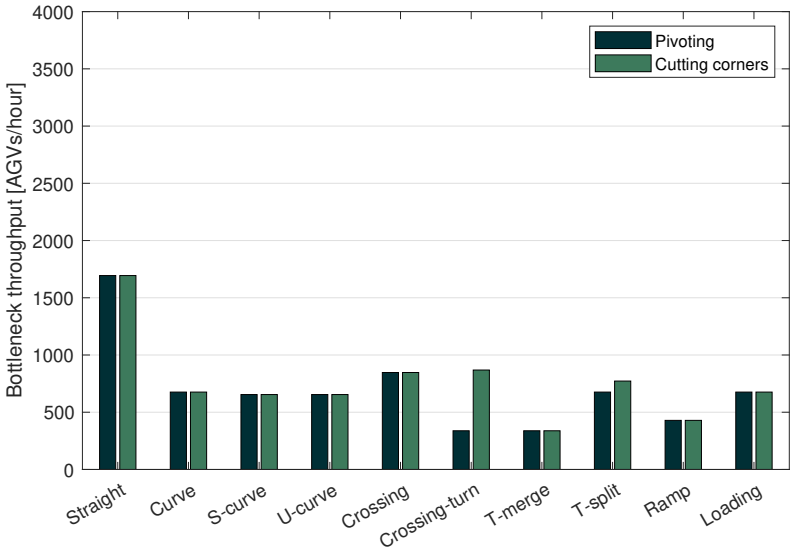


Figure 3.8: The bottleneck throughputs for benchmark scenarios for both turning by pivoting and by cutting corners.

It is apparent from this figure, that cutting corners only improves the 'Crossing-turn' scenario's bottleneck throughput. This is mainly as a result of AGVs not being able to claim enough tiles ahead in the other scenarios, thus not cutting the corners. From the figure, one can obtain which scenarios are actually potential bottlenecks, and which are not. It is apparent 'T-merge' and 'Ramp' are current culprits. 'Curve', 'S-curve', and 'U-curve' also have low throughputs.

3.5 Benchmark system configurations

The *AgvSorter* system is meant to be flexible, and should be applicable to different system configurations. Vanderlande operates in the airports, parcel, and warehouse industry. These market segments each require different types of layouts. In this project, we distinguish grid-type layouts and segment-type layouts. A grid-type layout is a layout where the majority of tiles has multiple incoming segments and multiple outgoing segments. A

segment-type layout is a layout where the majority of the tiles has one incoming segment and one outgoing segment. Two system configurations are introduced: a parcel sorting system, called *GridSorter*, representing grid-type layouts, and an airport baggage system, called *Airport*, representing segment-type layouts, see Figure 3.9 and 3.10, respectively. Also a hexagonal layout[3], designed to be the equivalent of the *GridSorter* layout, but with hexagonal tiles is introduced, which is called *Tessellation*, see Figure 3.11. In this thesis, these three system configurations are used as benchmarks to compare different versions of the *AgvSorter* control model, based on system throughput and the lead time percentile.

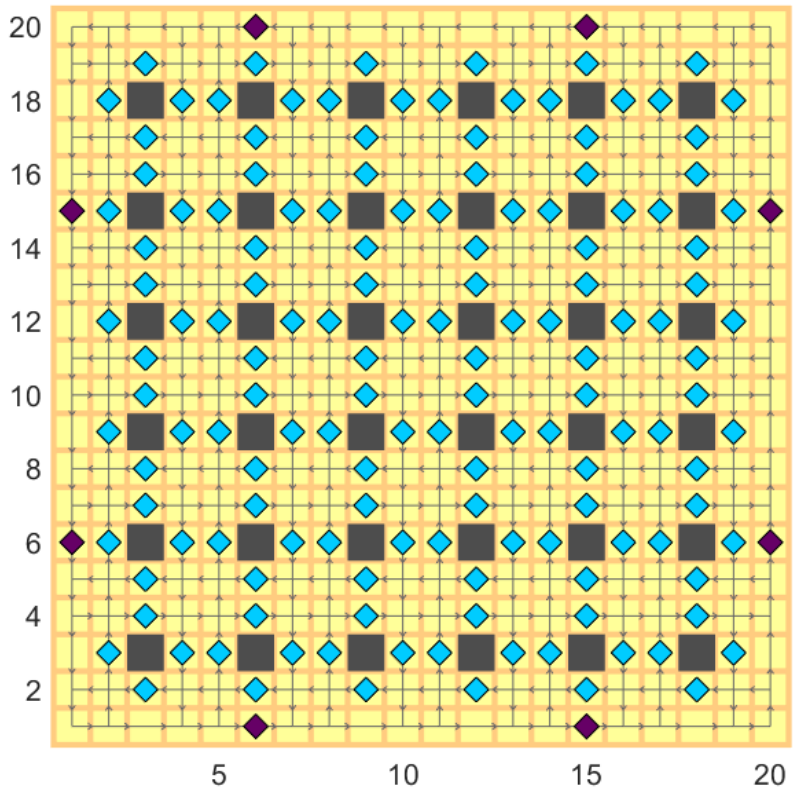


Figure 3.9: The *GridSorter* layout

The *GridSorter* layout has 20 x 20 tiles of size 1 x 1 m^2. It consists of 364 available tiles and 36 obstacles. There are eight sources and 144 destinations, four around each obstacle. The configuration parameters of *GridSorter* are listed in Appendix B.

[3]For the hexagonal grid layouts a patent is pending

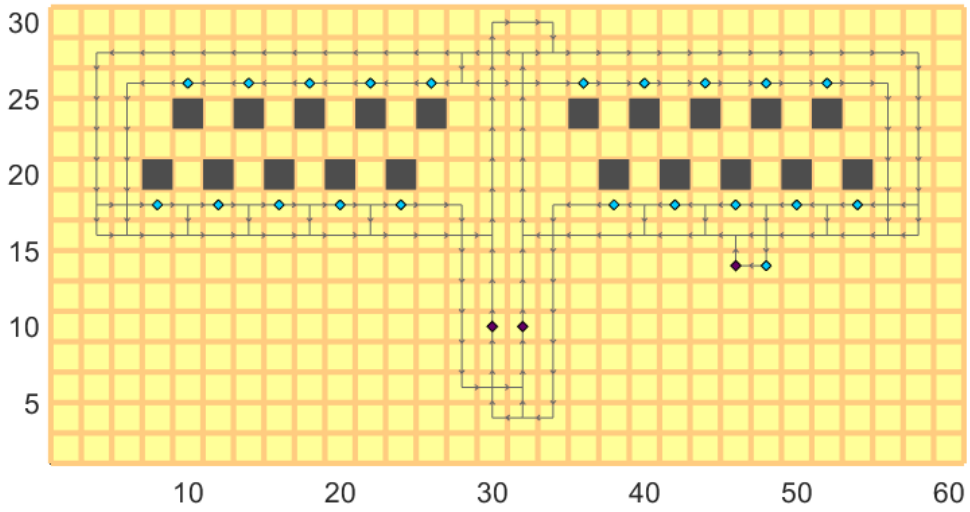


Figure 3.10: The *Airport* layout

The *Airport* layout has 15 x 30 tiles of size 2 x 2 m^2, although only 159 tiles have segments allocated to them. The layout further consists of 21 destinations, 3 sources, and ten obstacles. The configuration parameters of *Airport* are listed in Appendix B.

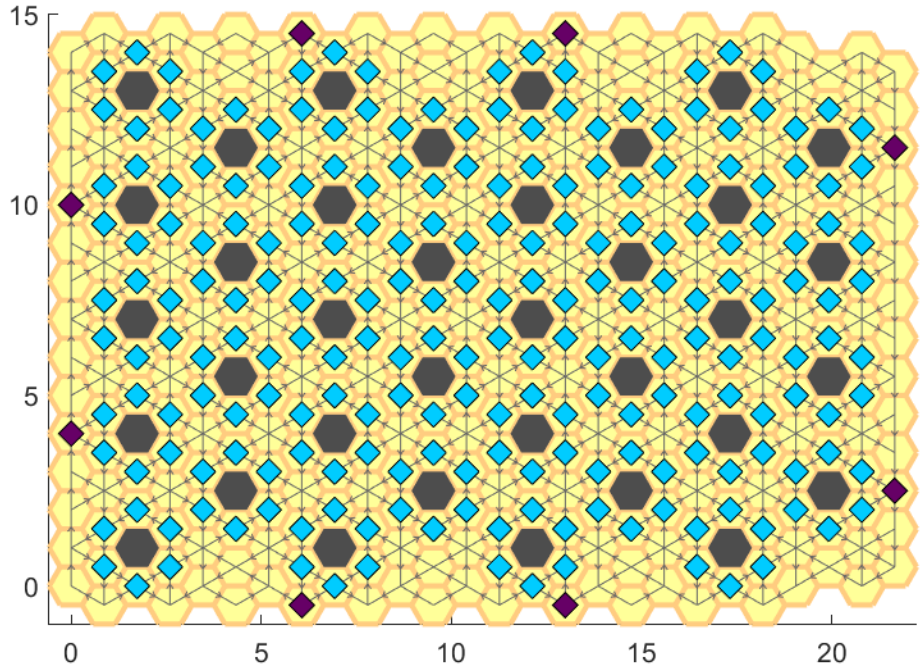


Figure 3.11: The *Tessellation* layout[4]

The *Tessellation* layout[4] has 15 x 26 tiles of height 1 m. It consists of 363 available tiles and 36 obstacles. There are eight sources and 216 destinations. The configuration parameters of *Airport* are listed in Appendix B.

[4]For the hexagonal grid layouts a patent is pending

3.6 Potential of platooning for grid-based AGV control

In section 3.3, the local behavior of a congested intersection was described, where one AGV is allowed on the tile alternately from different directions. Platooning would entirely change this behavior; instead an entire platoon would be able to cross the tile together, substantially reducing the time necessary to clear the congestion.

In section 3.4, the bottleneck throughputs of the baseline *AgvSorter* model were determined. Notable is the lack of improvement with the turning strategy cutting corners with respect to pivoting. This is explained by the fact that AGVs can often not cut a corner, because the AGV is not able to claim enough tiles ahead. This shows potential for a platooning strategy, where AGVs can cut a corner as a platoon, thus substantially increasing bottleneck throughputs.

A way of explaining the occurrence of delays in a grid-based system is that they are caused by the scarcity of unclaimed available tiles. AGVs reserve multiple tiles ahead, and as a result occupy more than one tile at a time. Platooning not only allows AGVs to drive at smaller inter-vehicular distance, but also allows AGVs to share tile reservations. It is expected that this will decrease the scarcity of unclaimed available tiles, as a string of n AGVs driving as a platoon occupies less space in the system, than a string of n AGVs not applying a platooning strategy.

3.7 Summary

The *AgvSorter* is controlled using grid-based control. Important tasks in grid-based control are task allocation, path planning and traffic control. This project focuses on traffic control. To gain insight in the *AgvSorter*, Vanderlande developed a discrete-event simulation model in MATLAB. To be able to compare different control models for the *AgvSorter*, benchmark scenarios and configurations are defined. Platooning is anticipated to bring a gain in performance with regard to the baseline model by reducing the time necessary to clear a congestion, improving bottleneck throughputs, and reducing scarcity of unclaimed available tiles.

4: Grid-based platooning

In the previous chapter, the *AgvSorter* model was described and analyzed. Additionally, the potential of platooning for the *AgvSorter* was described. In this chapter several platooning strategies, each one building onto the previous one, are designed for grid-based AGV systems.

4.1 Straight-line platooning

In the previous chapter, it was hypothesized that cutting a corner as a platoon would increase system performance. Before designing a strategy being able to cut a corner as a platoon, first a platooning strategy is designed for AGVs driving in a straight line: straight-line platooning.

4.1.1 Desired behavior

In order to achieve platooning, multiple vehicles need to accelerate and decelerate together, while maintaining a small inter-vehicular distance. This idea is contradicted by ground rule four of the baseline *AgvSorter* control model: *'Each tile can be reserved by one AGV at a time.'* In order to achieve platooning behavior, it should somehow be possible for AGVs to share a tile, while ensuring conflict-free driving. In Figure 4.1, it is illustrated what straight-line platooning would look like: two AGVs can drive on one tile.

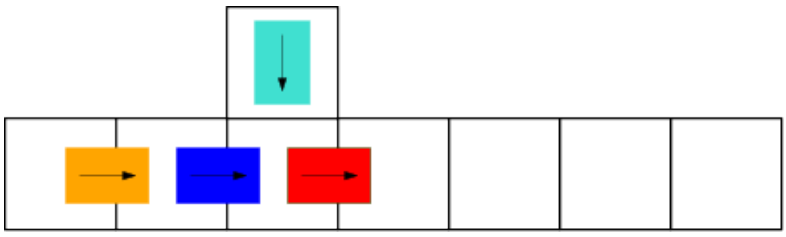


Figure 4.1: An illustration explaining the desired behavior of straight-line platooning

In the example, three AGVs (yellow, dark blue, and red) exhibit platooning behavior; a fourth AGV (cyan) can only reserve its next tile if this tile is unclaimed again. If the platoon reaches a turn, straight-line platooning should not allow the AGVs to make the turn as a platoon. Additionally, the straight-line platooning strategy should achieve the three main tasks of platoon control: ensure collision-free driving, minimize inter-vehicular distance, and guarantee string stability.

4.1.2 Strategy design

In designing the straight-line platooning strategy, it was chosen to establish a platooning behavior by creating a new dynamic between two preceding AGVs, named the leader-follower principle. In Chapter 3.3, two instigation mechanisms for tile reservation were described. The leader-follower principle is a new additional tile reservation mechanism. This mechanism is an interaction between two subsequent AGVs: the leader vehicle, and the follower vehicle. In Figure 4.2 two leader-follower pairs are shown.

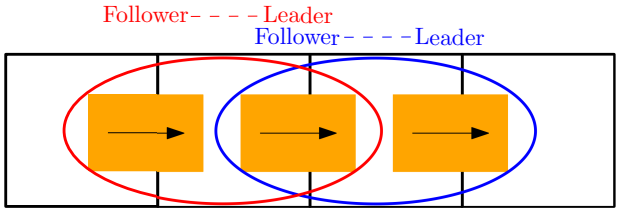


Figure 4.2: An illustration of leader-follower pairs

The workings of this mechanism are as follows: the leader AGV determines its subsequent reserved tiles, starting from its first reserved tile, for which it can indicate it is safe, from the leader's perspective, for the follower AGV to follow the leader. This decision is made using the indication conditions. The leader AGV gives an indication, after a small delay, for the subsequent reserved tiles that satisfy the indication conditions. This delay is a tuning parameter with which safety can be ensured, while minimizing inter-vehicular spacing. The indication conditions are stated and elaborated on next:

Indication conditions:

- *Reservation condition:* the leader has already reserved the next reserved tile on its path. This ensures no indication is given for tiles where the leader will come to a standstill.
- *Pivoting condition:* the leader is not currently pivoting. Pivoting takes relatively long, so if a leader is pivoting it cannot ensure it is gone before a potential follower enters the tile.
- *Leader turning point condition:* the leader has no turning point up to and including this tile. This condition is necessary, because straight-line platooning is not supposed to allow AGVs to make a turn as a platoon.

If a follower AGV wants to reserve the tile for which an indication is given, it can, if from the follower's perspective it is safe and optimal, follow-up on this indication. This means this AGV will try to take over the reservation of this tile from the leader vehicle, according to the flow chart presented in Figure 3.5. The decision if it is safe and optimal to follow-up on an indication for a given tile is made by verifying if all of the follow-up conditions are satisfied.

Follow-up conditions:

- *Indication condition:* the leader has made an indication for this tile.
- *Direction condition:* the follower enter this tile from the opposite segment the leader leaves this tile, because a follower vehicle should only follow-up if it is on the trail of the leader.
- *Speed condition:* the follower speed is not higher than the leader speed, so as to ensure the follower vehicle can never gain on its leader.
- *Maximum distance condition:* there is less than one tile distance between leader and follower (surface-to-surface), because vehicles should only perform a platooning action if they drive at a small inter-vehicular distance.
- *Follower turning point condition:* the follower does not need to turn before reaching this tile. Turning takes a relatively long time. In this time the leader is already driving. Performing a platooning action would result in an undesirably large inter-vehicular distance.

To maintain the established distance, a follower vehicle should not be allowed to drive at a higher speed than its leader vehicle. Therefore, as maximum segment speeds vary on the grid, the maximum allowed speed for a follower must be the minimum of (1) the maximum allowed speed of its leader, and (2) the maximum segment speed allocated to the AGV's current tile.

4.1.3 Resulting behavior

Figure 4.3 shows a situation in which the behavior prompted by straight-line platooning is apparent. In Appendix C, a more elaborate example with a platoon of three AGVs is presented.

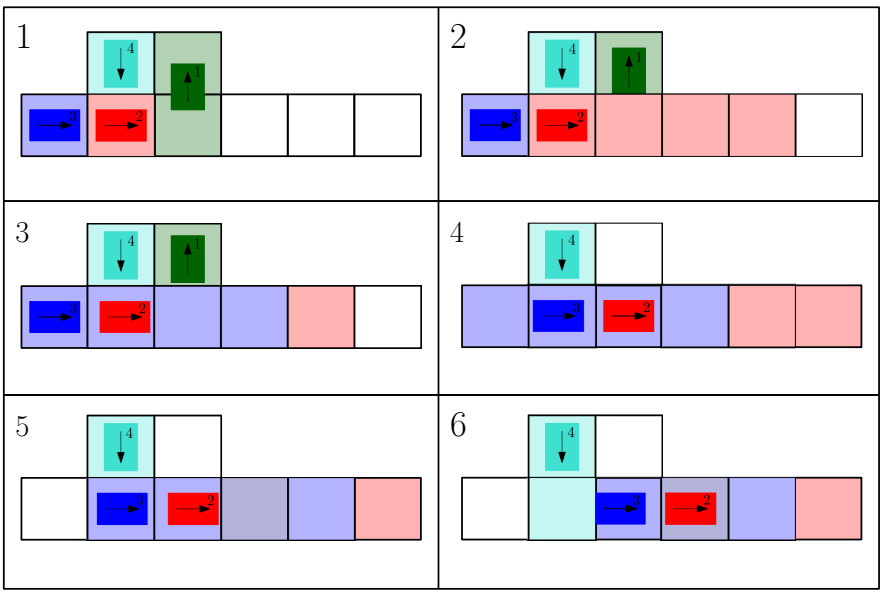


Figure 4.3: Behavior prompted by straight-line platooning

The figure consists of six snapshots, ordered chronologically:

1. AGV 1 (green) is occupying the next tile of AGV 2 (red), making AGV 2 have to wait.
2. After AGV 1 clears this tile, AGV 2 can reserve three tiles ahead.
3. After a small delay AGV 2 gives indications for its first three reserved tiles, and AGV 3 (dark blue) follows-up on these indications.
4. AGV 2 reaches the center of its next tile, and can make a new tile reservation.
5. After a small delay, AGV 2 gives an indication for its second-to-last reserved tile, and AGV 3 follows-up on this indication.
6. When AGV 3 clears the tile AGV 4 (cyan) has been waiting for, AGV 4 can reserve its next tile, and start moving.

This behavior corresponds to the desired behavior. A small inter-vehicular distance between platoon members is achieved, while precluding the occurrence of collisions. In the leader-follower principle, when having followed-up on the leader's indication, the follower does not react to perturbations of the velocity or position of the leader; thus, this principle results in string stability. Because a follower vehicle does not react to the leader's perturbations, to ensure safety in case of leader malfunction, the AGVs should be equipped with accurate on-board distance sensors and strong brakes. A negative side-effect of platooning is that long platoons can delay AGVs outside of the platoon substantially. Especially for segment-type layouts, where rerouting is often not possible, this is expected to have a negative impact on system throughput.

Straight-line platooning is designed to improve the benchmark scenarios 'Straight', 'Crossing', and 'Loading'. However, these scenarios already have relatively high bottleneck throughputs in the baseline. The throughput of segment-type layouts is hypothesised to be very dependent on the scenarios with the lowest bottleneck throughputs, as in these layouts rerouting is often not possible. Therefore, it is expected straight-line platooning has only a small positive effect on segment-type layouts. In grid-type layouts, AGVs have relatively much possible paths via different scenarios. The AGVs are distributed over the scenarios based on the capacity of these scenarios by smart path planning. Therefore, increasing any scenario's bottleneck throughput, not only the low, is expected to positively affect system throughput for grid-type layouts. Hence, straight-line platooning is anticipated to have a substantial positive effect on grid-type layouts.

4.2 Curved-line platooning

This section describes a platooning strategy designed to also cut a corner as a platoon: curved-line platooning. It builds further upon straight-line platooning. In Figure 4.4 the three types of turns applicable for cutting corners in rectangular layouts are shown. Three types of tiles are distinguished: entry tiles, turning tiles, and exit tiles. These types are respectively denoted by number 1, 2, and 3. The path separation point S is

the point at which the path associated with the pivoting strategy and cutting corners strategy start to differ.

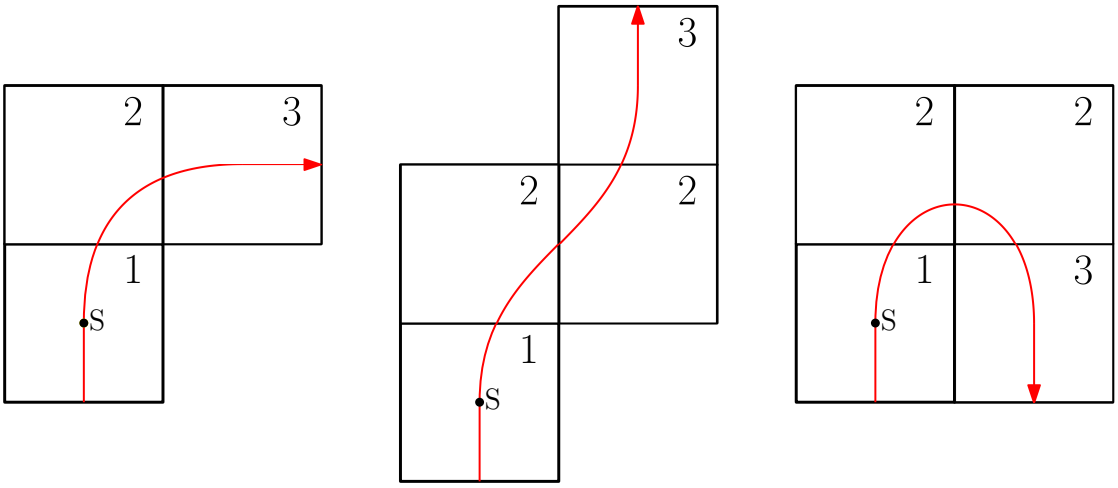


Figure 4.4: Tile definitions for different turns: entry tiles (1), turning tiles (2), exit tiles (3)

4.2.1 Desired behavior

Curved-line platooning should allow a platooning action if the leader vehicle cuts a corner. In Figure 4.5, all three possible paths for the follower where the leader cuts a corner are shown. In all three situations a curved-line platooning strategy should allow platooning behavior to occur.

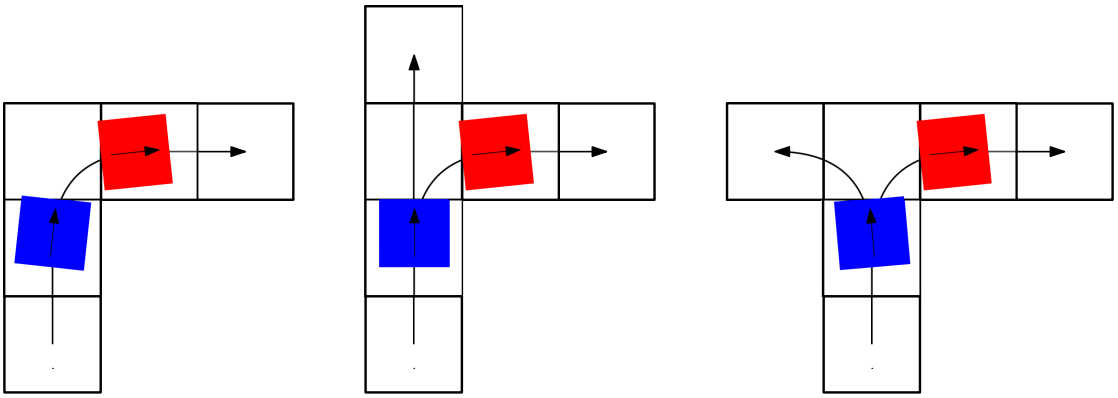
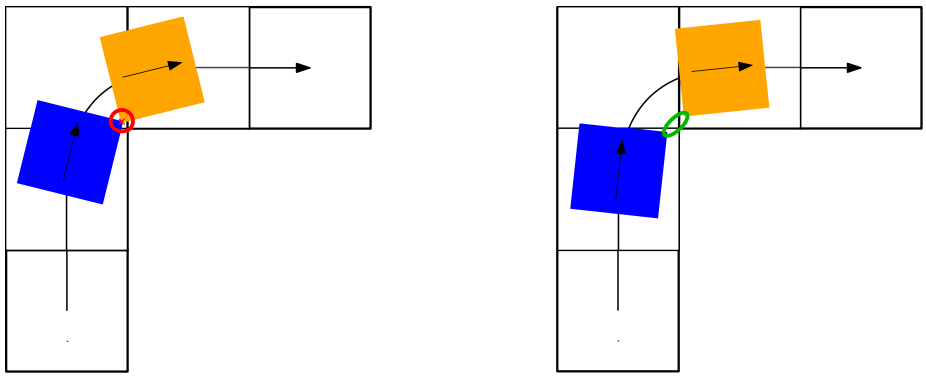


Figure 4.5: Three situations showing the desired behavior of cutting a corner as a platoon

It is required for the follower to maintain a safe inter-vehicular surface-to-surface (STS) distance. For rectangular tiles the desired minimum STS distance is $0.3 \cdot l_{\text{tile}}$. This is the STS distance two AGVs with worst case scenario geometry have when standing still at adjacent tiles, with l_{tile} being the length of the shortest side of a tile. The worst case scenario geometry of an AGV is so that it can exactly pivot on the spot within the borders of a tile. In Figure 4.5, the AGVs drive at this minimal distance.

4.2.2 Strategy design

Important questions in designing a curved-line platooning strategy are at which inter-vehicular center-to-center (CTC) arc distance, two consecutive AGVs fit through the corner as a platoon, and how to guarantee this distance. This distance is, from now on, denoted with d_{arc}. In Figure 4.6a two AGVs are shown that cut a corner at the most critical point, i.e. where the AGVs are closest to each other. The corner segments are approximated by circle pieces. In this situation, d_{arc} is equal to l_{tile}. As can be seen in the Figure, the AGVs do not fit through the corner with this value for d_{arc}. In Figure 4.6b, d_{arc} is enlarged with $0.25 \cdot l_{\text{tile}}$. Naturally, a safer situation results. The shortest STS distance between the AGVs at the critical point now is approximately the desired distance: $0.3 \cdot l_{\text{tile}}$. The minimal required value for d_{arc} therefore is $1.25 \cdot l_{\text{tile}}$.



(a): $d_{\text{arc}} = l_{\text{tile}}$: Collision

(b): $d_{\text{arc}} = 1.25 \cdot l_{\text{tile}}$: Safe distance

Figure 4.6: The effect of d_{arc} on the STS distance between AGVs when follower and leader drive at the critical point

The curved-line platooning strategy builds further upon the leader-follower principle. To achieve the earlier determined minimal inter-vehicular arc distance, if an AGV wants to follow-up on an indication for a tile associated with a (cutting) corner, a follower needs to pass a minimal distance check, before it is allowed to follow the leader. This check verifies if the value for $d_{\text{arc}} \geq 1.25 \cdot l_{\text{tile}}$ when the leader AGV drives at the critical point. The critical point for the leader and follower are shown by the yellow and blue AGV in Figure 4.6b, respectively. This check is referred to as the minimum distance check. In Appendix D this check is elaborated on.

Similar to straight-line platooning, the leader may only give an indication after a delay when all the indication conditions are satisfied. The indication and follow-up conditions for curved-line platooning need to be adjusted to allow AGVs to cut a corner as a platoon. The conditions differ based on the tile of interest. The adjusted conditions are stated and elaborated on next. If the condition differs to the corresponding condition for straight-line platooning, a more detailed elaboration is given.

Indication conditions:

- If the tile of interest is an entry tile, or the first corner tile in an S-turn or U-turn:
  - *Reservation condition:* the leader has already reserved its next two tiles. It is necessary that the next two tiles are reserved, because an AGV cutting a corner can at one time occupy three tiles.
  - *Pivoting condition:* The leader is not currently pivoting.
  - *Leader turning point condition:* The leader has no pivoting turning point up to and including this tile.
- Any other tile:
  - *Reservation condition:* The leader has already reserved its next tile.
  - *Pivoting condition:* The leader is not currently pivoting.
  - *Leader turning point condition:* The leader has no pivoting turning point up to and including this tile.

Follow-up conditions:

- If the tile of interest is a corner tile, or exit tile:
  - *Indication condition:* the leader has indicate a release for this tile.
  - *Direction condition:* the follower enters the tile from the opposite segment the leader leaves this tile.
  - *Speed condition:* the follower speed is not higher than the leader speed.
  - *Minimum distance condition:* the minimum distance check is described earlier in this chapter, and ensures a safe distance between leader and follower vehicle.
  - *Maximum distance condition:* there is less than one tile distance between leader and follower (surface-to-surface).
  - *Follower turning point condition:* the follower does not need to pivot before reaching this tile.
- Any other tile:
  - *Indication condition:* the leader has indicated a release for this tile.
  - *Direction condition* the follower enters the tile from the opposite segment the leader enters this tile.
  - *Speed condition:* the follower speed is not higher than the leader speed.
  - *Maximum distance condition:* there is less than one tile distance between leader and follower (surface-to-surface)
  - *Follower turning point condition: the follower does not need to pivot before reaching this tile.*

4.2.3 Resulting behavior

Figure 4.7 shows a situation in which the behavior prompted by curved-line platooning is apparent.

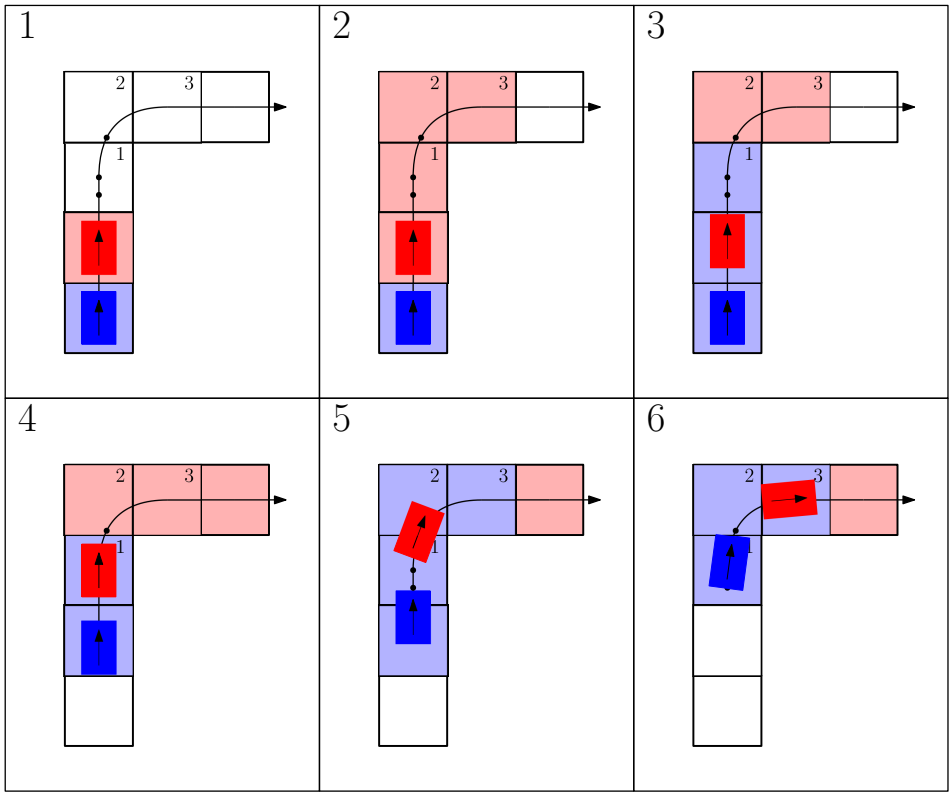


Figure 4.7: Behavior prompted by curved-line platooning

The figure consists of six snapshot, ordered chronologically:

1. Both the leader (red) and follower (blue) are at a standstill.
2. The leader can reserve three tiles ahead.
3. After a short delay the leader gives an indication for its first three tiles. The follower gives a follow-up on two of these indications. Because the minimum distance condition is not satisfied, the indication on the third tile, which is a turning tile, can not be followed-up.
4. The leader reaches the entry tile, and can reserve a new tile. At the same time the leader gives a new indication.
5. The follower had to slow down, because it had not yet reserved its next tile. While the AGV is slowing down, the minimum distance condition is satisfied, and the AGV can follow-up on all given indications.
6. At the critical point the distance between the leader and follower AGV is sufficient, and the platoon can safely pass the turn.

This behavior corresponds to the desired behavior. Platooning actions are also allowed if the leader vehicle cuts a corner, and a small, but safe distance between platoon members is guaranteed. Similarly to straight-line platooning, the leader-follower principle results in string stability.

4.3 Lenient cutting corners

Curved-line platooning only occurs if the leader vehicle decides to cut a corner. However, the baseline cutting corners is conservative in deciding to cut a corner or not; if not all tiles associated with the turn are reserved by the AGV, the AGV will not cut the corner. Lenient cutting corners is an alternative solution for the baseline cutting corners, less conservative in deciding to cut a corner. The reason for the baseline cutting corners to be this conservative is visualized in the most left illustration in Figure 4.8.

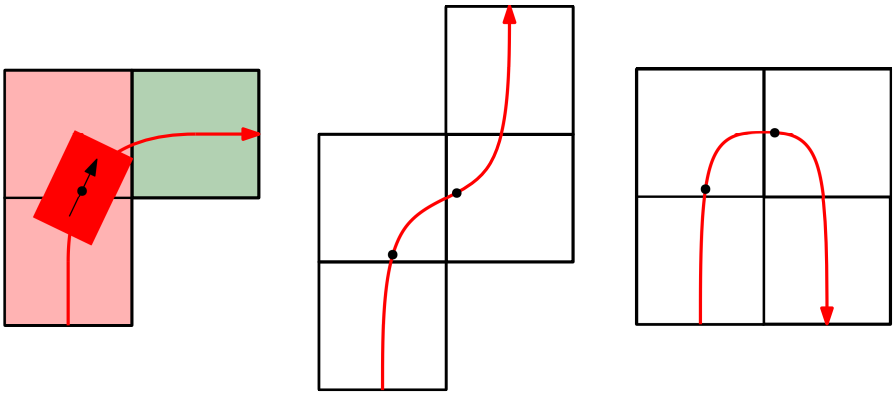


Figure 4.8: The cutting corners waiting points

The illustration shows an AGV that already decided to cut a corner without having reserved the exit tile, and is waiting at the black dot: the cutting corners waiting point. This point is defined as the last point on the AGV's path where the AGV does not occupy the unreserved tile. S-turns and U-turns have two waiting points. The location of the waiting points is indicated in Figure 4.8 for all three types of turns with a black dot. While waiting at the waiting point, the AGV occupies two tiles, until the next tile becomes free. If this takes only a small time, this is not really an issue. However, if the exit tile remains claimed for a long time, it would have been advantageous for the AGV to wait and pivot at the center of the corner tile instead. In conclusion, it is relevant to design a more lenient cutting corners strategy, so that the first AGV in a platoon will more often decide to cut a corner, without this AGV indefinitely occupying two tiles. The remainder of this subsection describes the desired behavior, strategy design, and resulting behavior of the lenient cutting corners strategy, respectively.

4.3.1 Desired behavior

The behavior that the lenient cutting corners strategy should bring about, is that single and leader AGVs should more often cut a corner, without AGVs occupying two tiles for too long.

4.3.2 Strategy design

The lenient cutting corners strategy handles when to turn by pivoting or by cutting corners. For this strategy a cutting corners decision point is defined. This is the point before which an AGV has to have made the decision to turn by pivoting or by cutting corners. By default the AGV assumes a pivoting strategy, but if on or before reaching the decision point certain conditions are satisfied, the corner is cut. The position of this point is defined as the first encountered point of the following points:

1. the point where the AGV needs to start decelerating to come to a standstill at the cutting corners waiting point.
2. the point where the AGV needs to start decelerating to reach the minimum of the maximum segment speeds of the two possible paths at the path separation point.

The locations of these points are dependent on the AGV's speed, not on the maximum speed. This means the decision point is not a fixed point, but can vary. At standstill, the decision point corresponds with the path separation point. An example of the position of the cutting corners decision point with respect to the cutting corners waiting point and path separation point is given in Figure 4.9.

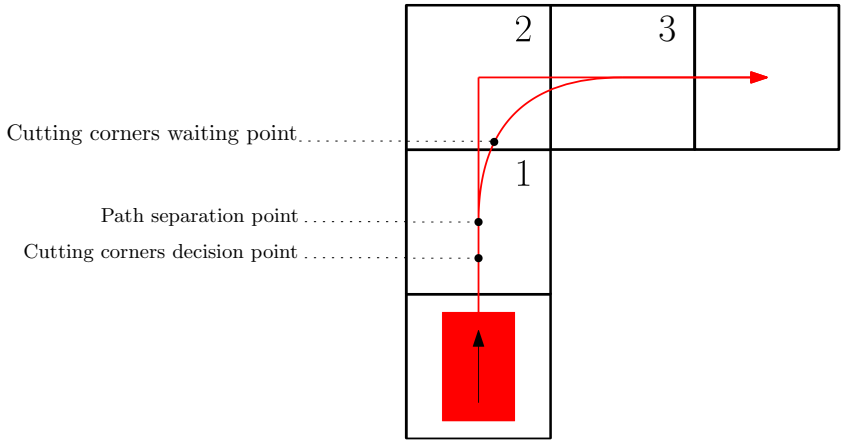


Figure 4.9: An example of where the cutting corners waiting point, path separation point, and cutting corners waiting point are located.

For the lenient cutting corners strategy it must be defined under which conditions the AGV may cut the corner. The condition for the baseline cutting corners strategy required the AGV to have reserved the entry, turning, and exit tiles. The lenient cutting corners strategy allows an AGV to cut the corner if both of the following, more lenient conditions are satisfied:

- The entry tile and first turning tile must be reserved by the AGV.
- The second turning tile (only present in S-turn and U-turn) and exit tile must satisfy either of the following three conditions:
  1. Be reserved by the AGV.
  2. Be unclaimed, and the AGV is allowed to reserve the tile next.

3. Be reserved by another AGV, who already reserved its next tile, and the AGV is allowed to reserve the tile next.

In verifying if the AGV is allowed to reserve the next tile, its position on the reservation candidate list should be considered, as well as deadlocks and grouped claims. For easier implementation and to preserve the possibility for the follower vehicle to reroute if the leader malfunctions, it can be considered to have the cutting corners waiting point coincide with the path separation point; the path separation point is the last point from which a different route can still be chosen. However, in a perfect world, without AGVs malfunctioning, this reduces bottleneck throughputs. Therefore, it was decided to not change the location of the cutting corners waiting point.

4.3.3 Resulting behavior

Figure 4.10 shows four situations in which the behavior prompted by the lenient cutting corners strategy is apparent.

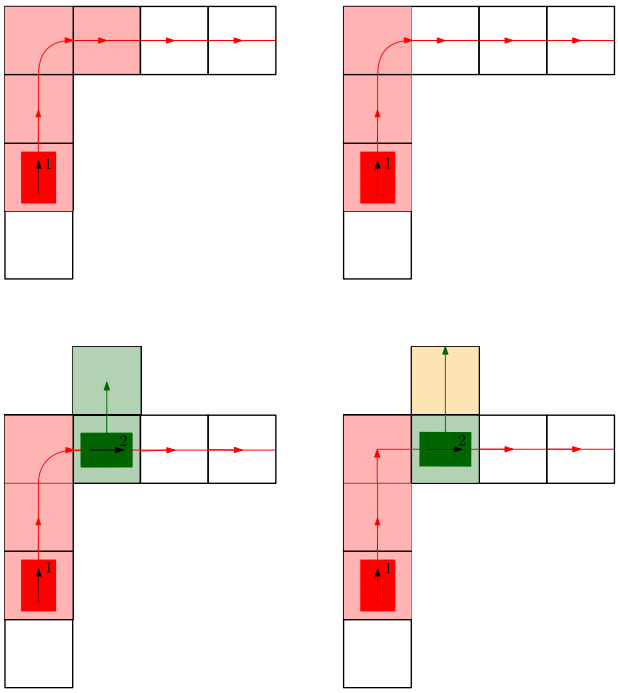


Figure 4.10: Behavior prompted by lenient cutting corners

The first situation (left upper corner) shows the situation where all tiles associated with the corner are reserved. As a result AGV 1 (red) is allowed to cut the corner, as is illustrated by the path. The second situation (right upper corner) shows the situations where the entry tile and turning tile are reserved, but the exit tile is not. AGV 1 is allowed to reserve the tile next, therefore the conditions for cutting a corner are satisfied. Thus, the corner may be cut by the AGV. The third situation (left lower corner) shows a situation where the entry tile and first turning tile are reserved by AGV 1. The exit tile is still claimed by AGV 2 (green). However, as AGV 2 has already reserved its next tile, AGV 1 is allowed to cut the corner. The fourth situation (right lower corner) shows a

situation where the exit tile is reserved by AGV 2, but AGV 2 has not yet reserved its next tile. As a result, AGV 1 is not allowed to cut the corner. Compared to the baseline cutting corners strategy, this is an improvement, as the baseline cutting corners strategy would only allow the AGV to cut the corner in the first situation.

It can be concluded that the lenient cutting corners strategy will more often allow AGVs to cut a corner, than the baseline cutting corners strategy. The strategy still does not allow AGVs to wait at the cutting corners waiting point indefinitely, while claiming two tiles. This corresponds to the desired behavior.

4.4 Platoon coordination

Until now the focus has been on the behavior of one platoon only; what happens when platoons meet other platoons, or single AGVs has yet been neglected. Long platoons can delay AGVs outside of the platoon significantly, see Figure 4.11.

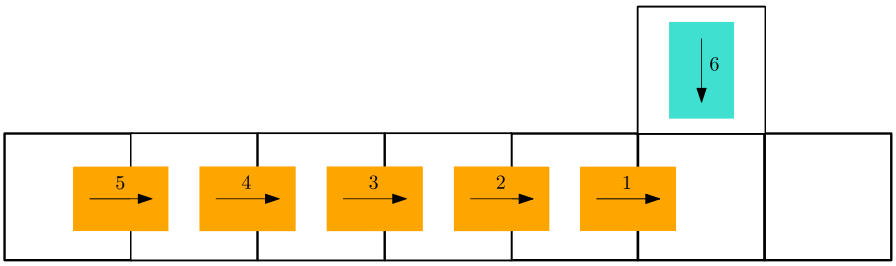


Figure 4.11: A single AGV delayed by a platoon of five AGVs

In this figure, the numbers denote the order in which the AGVs are allowed to cross the crossing. The single AGV has to wait for all five platoon members to pass before it can continue. As there is no limit on the length of a platoon, platoons can cause long delays for AGVs outside of the platoon. This type of behavior can negatively impact system performance. Therefore, platoon coordination is necessary to prevent long delays. Applying a platooning strategy without coordination rules is from now on referred to as unrestricted platooning. The remainder of this section first describes the desired behavior for the coordination rules. Then the design of three coordination strategies is explained, followed by the resulting behavior of these strategies.

4.4.1 Desired behavior

The behavior that should be brought about by the coordination rules, is that at a crossing, the number of platoon members an AGV outside of the platoon has to wait for is bounded. For how many platoon members this AGV has to maximally wait, should be a pre-defined system parameter.

4.4.2 Strategy design

Realising the desired behavior can be achieved by various coordination strategies. Three different strategies are designed: a permitted platoon length strategy, which limits platoon

size, and two variations on a right-of-way (RoW) platoon length strategy, which both break up platoons to allow AGVs outside of the platoon to cross.

Permitted platoon length

A first possibility achieve the desired behavior, is to limit the length of a platoon, by setting a maximum permitted platoon length. This maximum permitted platoon length is a restriction on the amount of subsequent AGVs in a platoon crossing a tile.

Right-of-Way platoon length

The rule based on the permitted platoon length sometimes unnecessarily breaks up a platoon; it is not necessary to break up a platoon, if no other AGV is waiting on the same tile. Therefore, the maximum Right-of-Way (RoW) platoon length is introduced. This is the maximum platoon length at which a platoon has RoW over AGVs outside of the platoon. This concept is explained using Figure 4.12.

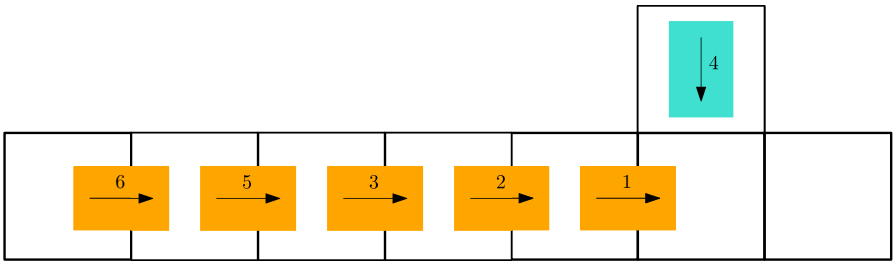


Figure 4.12: A visualization of the concept of Right-of-Way platooning, with a maximum RoW platoon length of three

In this Figure, the numbers denote the order in which the AGVs are allowed to cross the crossing. The platoon of five AGVs breaks up at the crossing after three AGVs, so that the single AGV may pass. For this strategy to work, it is necessary to count the number of subsequent AGVs passing a tile. Two variations are introduced: RoW true platoon length and RoW observed platoon length.

- *RoW true platoon length strategy:* the first, most intuitive, moment to start counting is when the first member of the platoon enters the tile of interest: the true platoon length. A RoW platoon length strategy based on the true platoon length means sometimes an AGV immediately has RoW over a platoon if the count is already high on arrival at the crossing, and sometimes it needs to wait for some platoon members to cross first.

- *RoW observed platoon length strategy:* the platoon count is started if an AGV outside of the platoon (the observer) also wants to reserve the tile of interest: the observed platoon length. The observed platoon length is always equal to or smaller than the true platoon length. A RoW platoon length strategy based on the observed platoon length means an AGV, upon arrival at a crossing, always needs to wait for the specified maximum RoW platoon length before having RoW over a platoon.

We have introduced three different coordination rules: permitted platoon length, true RoW platoon length, and observed RoW platoon length. Multiple rules can 'coexist' in a system, however, only under the following condition:

Maximum permitted platoon length $\geq$ *Maximum RoW true platoon length* $\geq$ *Maximum RoW observed platoon length*

These conditions are necessary for the different rules to not be obsolete, as the permitted platoon length strategy is more restrictive than the RoW true platoon length strategy, and the RoW true platoon length is more restrictive than the RoW observed platoon length.

4.4.3 Resulting behaviors

Next, the resulted behavior for the three newly introduced rules is elaborated on.

Permitted platoon length

The permitted platoon length strategy limits the amount of subsequent platoon members crossing any given tile, thus also limiting the amount of platoon members for which an AGV outside of the platoon needs to wait. Nevertheless, this strategy does temporarily allow platoons longer than the maximum permitted platoon length, see Figure 4.13.

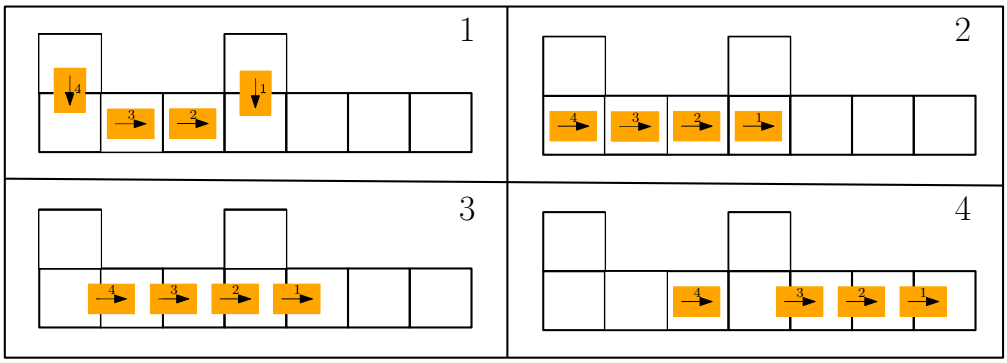


Figure 4.13: The behavior prompted by the permitted platoon length strategy, with a maximum permitted platoon length of three.

In the figure, the maximum permitted platoon length is three. From snapshots 1 through 3, it is evident that longer platoons can occur. In snapshot 4, it can be seen that AGV 4 cannot perform a platooning action for its next tile, as otherwise the restriction on the maximum permitted platoon length would be violated. Therefore, the platoon must be broken up.

RoW true platoon length

Figure 4.14 shows the situation in which the behavior prompted by a RoW true platoon length strategy is evident.

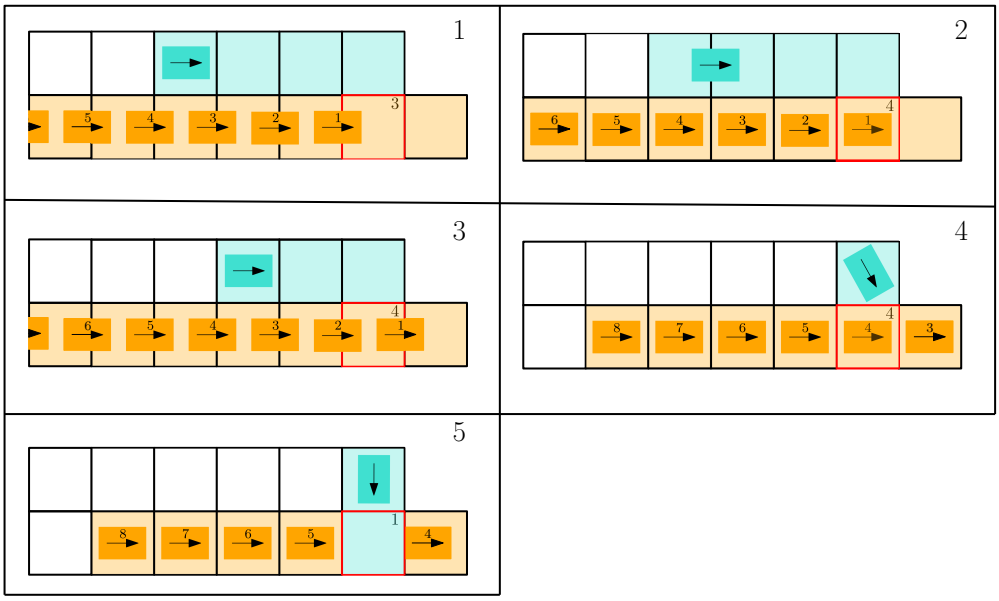


Figure 4.14: The behavior prompted by the RoW true platoon length strategy, with a maximum RoW true platoon length of three

The figure consists of five snapshots, ordered chronologically. The number in the upper right corner of the highlighted tile denotes the true platoon length count for this tile.

1. All AGVs have reserved three tiles ahead. The red bordered tile therefore has a count on the true platoon length of three.
2. The highlighted tile gets reserved by AGV 4, as it is still out of reach from the single AGV (cyan). The count goes to four.
3. The highlighted tile is in reach of the cyan AGV. The single AGV has RoW, hence no new platooning actions are allowed.
4. The single AGV is pivoting in anticipation to go to the highlighted tile, and AGV 5 comes to a standstill.
5. AGV 4 leaves the tile, and the single AGV can reserve the highlighted tile.

The platoon is broken up relatively quickly for the single AGV. The AGV can, almost without waiting, continue its path.

Observed RoW platoon length

Figure 4.14 shows a situation in which the behavior prompted by a RoW observed platoon length strategy is evident.

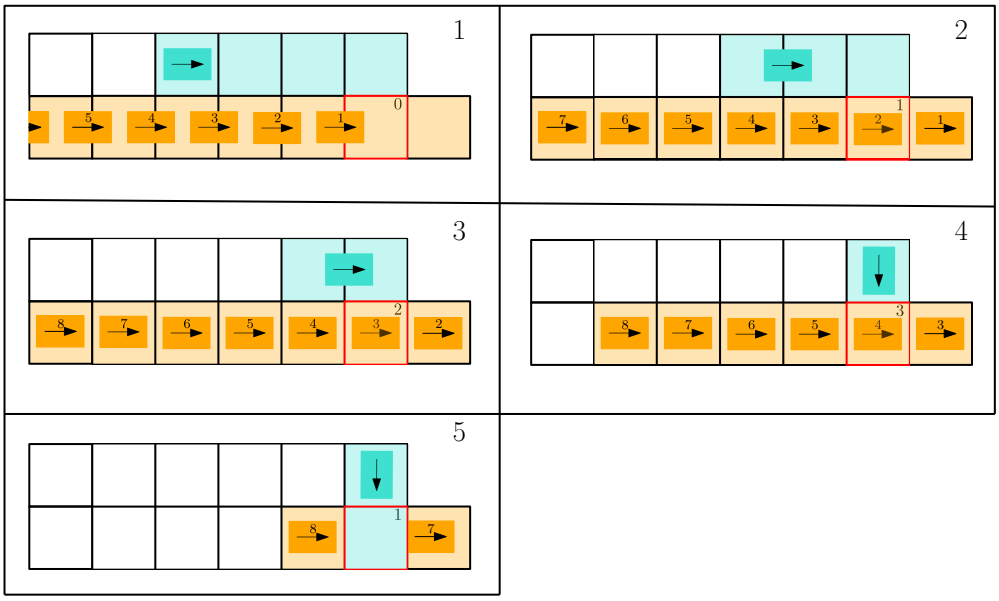


Figure 4.15: The behavior prompted by the RoW observed platoon length strategy, with the maximum RoW observed platoon length is three

The figure consists of five snapshots, ordered chronologically. The number in the upper right corner of the highlighted tile denotes the observed platoon length.

1. All AGVs have reserved three tiles ahead. The highlighted tile has a count on the observed platoon length of zero, as the tile is still out of reach from the cyan AGV.
2. The first reservation is made by a member of the platoon (AGV 5), while the highlighted tile is in reach of the single AGV (cyan). The observed platoon length count goes to one.
3. AGV 6 reserves the highlighted tile via a platooning action, and the observed platoon length count goes to two.
4. AGV 7 reserves the highlighted tile via a platooning action, and the observed platoon length count goes to three. The maximum RoW observed platoon length is reached. Giving the single AGV RoW from now on.
5. AGV 7 leaves the highlighted tile, and the single AGV can reserve the tile.

The single AGV has to wait for AGV 7 before it can continue. In perspective, with the same start situation for the RoW true platoon length strategy, the single AGV was allowed after AGV 4.

All three discussed rules satisfy the desired behavior. Which rule, or combination of rules, results in the best system performance is hard to predict without empirical analyses. It is expected that the coordination rules counteract the platooning side effect of AGVs having to wait for long platoons. Earlier, it was hypothesized that this side effect would especially impact segment-type layouts. Coordination rules are therefore anticipated to especially positively affect segment-type layouts.

4.5 Summary

This chapter described several strategies for, or associated with platooning. Straight-line platooning is a platooning strategy for AGVs driving in a straight line. This strategy does not allow to cut a corner as a platoon. Therefore, curved-line platooning is developed. Curved-line platooning occurs only if a leader vehicle decides to cut a corner. As the baseline cutting corners is conservative in deciding whether to cut a corner or not, a more lenient strategy was developed: lenient cutting corners. Lastly, several platooning coordination strategies were developed, to handle platoons meeting other platoons or single AGVs.

5: Implementation

In the previous chapter, several strategies for, or associated with platooning are described. For simulation analyses on the benchmark system configurations, straight-line platooning and all platoon coordination strategies are implemented in the *AgvSorter* MATLAB simulation model. Due to the current implementation of the simulation model, curved-line platooning and lenient cutting corners are not implemented. This chapter focuses on verifying, and validating the implementation of straight-line platooning and the coordination rules in the *AgvSorter* MATLAB simulation model.

5.1 Verification of grid-based platooning implementation

To determine whether straight-line platooning and the coordination rules are implemented correctly, several verification tests are performed.

The simulation model comes with a visualization model, as described in Chapter 3. This allows to follow the movements of all the AGVs. Using the visualization it is verified that the desired (local) behaviors are achieved by the designed strategies.

In addition to verifications performed with the visualization model, several test cases are built. For straight-line platooning, it is verified whether the leader vehicle gives indications in the correct situations. It is verified that an AGV gives an indication if and only if the indication conditions are satisfied, and that an AGV can give indications multiple tiles ahead. For the follower vehicle it is verified that it follows-up on an indication if and only if the follow-up conditions are satisfied.

For the permitted platoon length strategy it is verified that only platoons smaller or equal to the max permitted platoon length cross a tile. For the true RoW platoon length strategy, it is verified that AGVs outside of a platoon have RoW if the true platoon length count at the tile exceeds the maximum RoW true platoon length. Similarly, for the observed RoW platoon length strategy, it is verified that AGVs outside of a platoon have RoW if the observed platoon length count at the tile exceeds the maximum RoW observed platoon length.

5.2 Validation of grid-based platooning implementation

The *AgvSorter* simulation model is still in development. Several aspects of the model do not correspond to a realistic system yet. Firstly, in the simulation model, AGVs never run out of battery, thus not requiring charging, and never malfunction. Secondly, in the current model, there is a continuous inflow of items at the sources, and an item can always be immediately processed at a destination. This is not realistic, as the processes and queues that serve as the *AgvSorter*'s interfaces with the outside world are ignored. Thirdly, communication between the central controller and AGVs is lossless and without delay.

The simulations therefore provide a higher throughput than would actually be realistic. However, if one approach performs better than another in the simulation model, it is also expected to perform better in reality. Therefore, this model is sufficient for this thesis, in which different traffic control strategies are analyzed and compared. For this reason, all simulation results are presented relative to the baseline *AgvSorter* model.

6: Results & discussion

This chapter presents the results of the strategies presented in Chapter 4. The results are focused on the KPI's defined in Chapter 1, relative to the baseline *AgvSorter* system. First, the strategies are compared using the benchmark scenarios. Subsequently, the results of simulations performed with the benchmark system configurations for the different strategies are presented and discussed. The influence of the indication delay parameter, and the effect of grid-based platooning on the quality attributes are also analyzed.

6.1 Benchmark scenario results

In Chapter 3, the benchmark scenarios are introduced, and for the baseline *AgvSorter* model the local bottleneck throughputs are calculated for both pivoting and cutting corners. In this section also the benchmark scenario bottleneck throughputs are presented for the *AgvSorter* model with straight-line platooning, and with curved-line platooning. The calculations are shown in Appendix E.

Table 6.1 shows the effect of grid-based platooning strategies on the benchmark scenarios. It is denoted with a '+' which strategy increases the bottleneck throughputs of which scenarios. No bottleneck throughput decreases by applying a grid-based platooning strategy. A '+/−' denotes an increased bottleneck throughput, accompanied by a negative side effect, which is elaborated on later. For curved-line platooning, a '≀' denotes the same bottleneck throughputs are achieved as for straight-line platooning.

	Straight	Curve	S-curve	U-curve	Crossing	Crossing-turn	T-merge	T-split	Ramp	Loading
Straight-line platooning	+				+				+/−	+
Curved-line platooning	≀	+	+	+	≀	+	+	+	+	≀

Table 6.1: Effect of straight-line platooning and curved-line platooning on the bottleneck throughputs of the benchmark scenarios

In Figure 6.1 the bottleneck throughputs of the benchmark scenarios are shown for the different strategies.

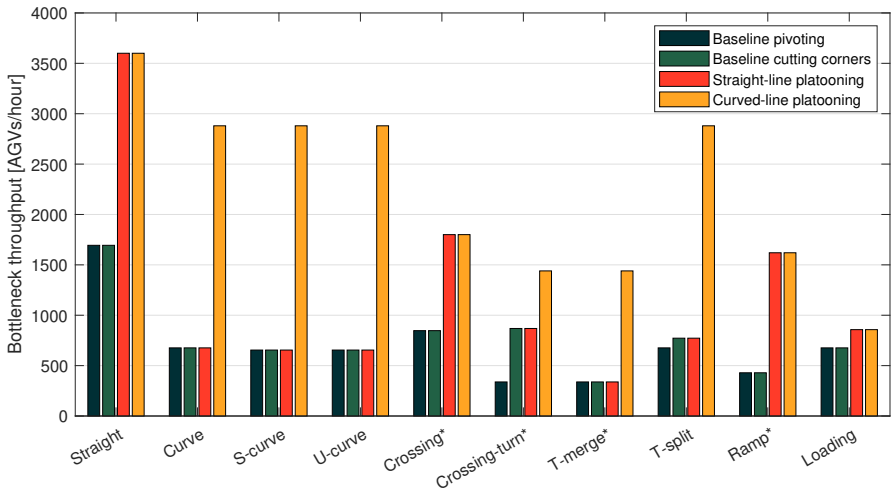


Figure 6.1: Throughputs benchmark scenarios

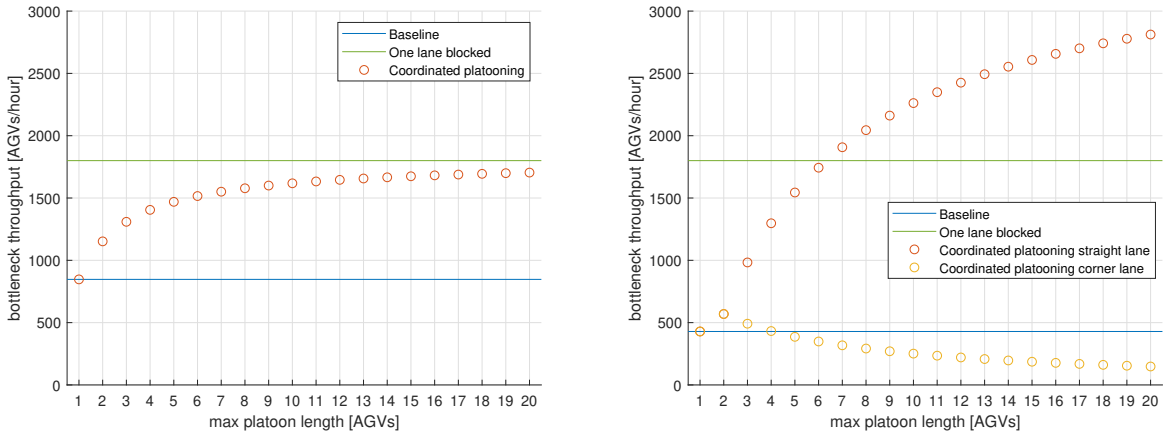
From Figure 6.1, it is obtained that baseline cutting corners, for most scenarios, gives no improvement to the bottleneck throughputs with respect to baseline pivoting. This is caused by AGVs not being able to reserve enough tiles ahead in case of push behavior, thus not cutting the corner.

Straight-line platooning only impacts scenarios where at least some AGVs drive in a straight lane, as can be seen in Table 6.1. In Figure 6.1, we see an increased bottleneck throughput for these scenarios. For 'Straight' the most significant improvement occurs: from 1694 AGVs/hour to 3600 AGVs/hour. Also, the peak performance of 'Loading' is improved with respect to the baseline strategy from 676 AGVs/hour to 857 AGVs/hour. 'Crossing' and 'Ramp' are marked with an asterisk, because these scenarios require some extra explanation. In these scenarios two AGV flows have to merge, thus their bottleneck throughputs are dependent on the coordination rules. The throughputs shown in the figure for these scenarios represent the average throughput of the two incoming lanes if the AGV flow of one lane is blocked. Therefore, this represents an overestimation of the real bottleneck throughputs. Later in this section this is further elaborated on.

Curved-line platooning additionally impacts scenarios where AGVs have to make a turn: 'Curve', 'S-curve', 'U-curve', 'Crossing-turn', 'T-merge', 'T-split', and 'Ramp'. The effect of curved-line platooning is also shown in Figure 6.1. For 'Curve', 'S-curve', and 'U-curve' the bottleneck throughput substantially increases from 676 to 2880 AGVs/hour, and for 'T-split' from 772 to 2880 AGVs/hour. 'Crossing-turn', 'T-merge', and 'Ramp' are marked with an asterisk, as these scenarios also have multiple incoming lanes. These scenarios are treated similarly to the earlier mentioned scenarios marked with an asterisk.

Scenarios with multiple incoming lanes

The bottleneck throughput of scenarios with multiple incoming lanes is dependent on the coordination rules. Switching from one lane to another leaves more inter-vehicular spacing, thus reduces bottleneck throughput. Both lanes, however, need to steadily flow to not cause large delays for individual AGVs. In Figure 6.2, for 'Crossing', and for 'Ramp' the effect of straight-line platooning with a maximum platoon length on the bottleneck throughput is investigated. The maximum platoon length denotes the amount of subsequent AGVs after which is switched from one lane to another.



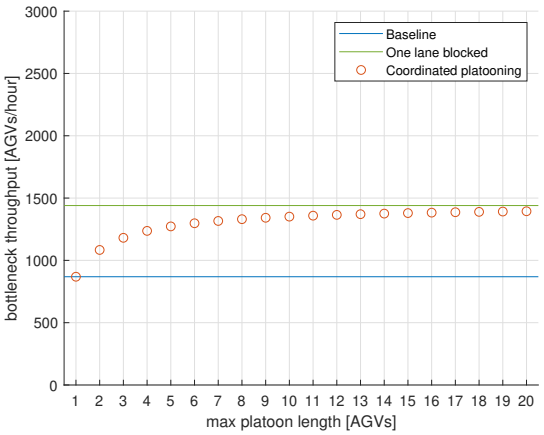
(a): Crossing **(b):** Ramp

Figure 6.2: Bottleneck throughputs with straight-line platooning for scenarios with two incoming lanes, varying the maximum platoon length

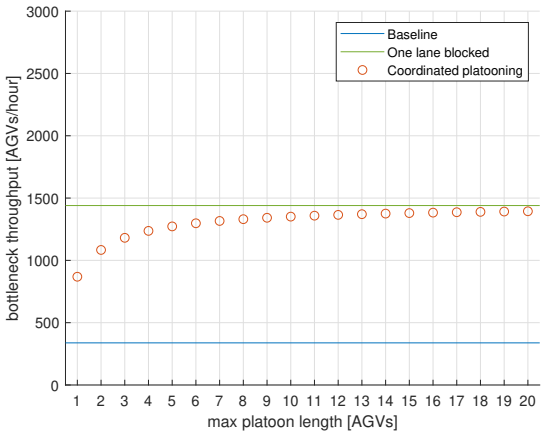
In Figure 6.2a, the 'Coordinated platooning' graph represent the bottleneck throughputs with varying *max platoon length* for the scenario 'Crossing'. For reference, the baseline bottleneck throughput, and the average bottleneck throughput if one lane is blocked, are also shown. The higher the *max platoon length*, the higher the bottleneck throughput, as there is less switching between lanes. This increase flattens out, approaching the 'One lane blocked' graph for larger *max platoon lengths*.

In Figure 6.2b, the AGV flows are visualized for the scenario 'Ramp'. This scenario is asymmetrical: one incoming lane, the straight lane, does allow straight-line platooning, and the other, the corner lane, does not. This results in a lower bottleneck throughput for the corner lane. One AGV from the corner lane may pass for every platoon from the straight lane. The resulting bottleneck throughputs for the straight lane and the corner lane are visualized with the 'Coordinated platooning straight line' graph and the 'Coordinated platooning corner lane' graph, respectively.

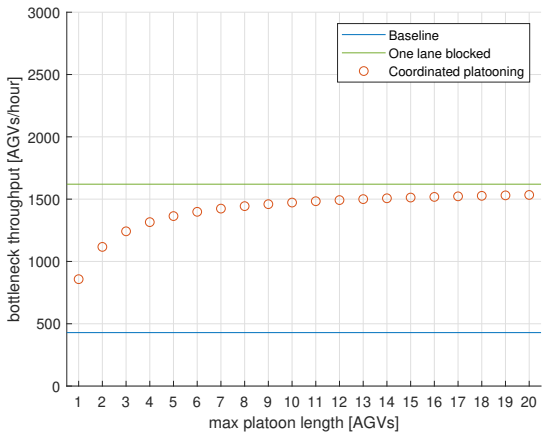
In Figure 6.3, for 'Crossing-turn', 'T-merge', and 'Ramp' the effect of curved-line platooning with a maximum platoon length on the bottleneck throughput is investigated.



(a): Crossing-turn



(b): T-merge



(c): Ramp

Figure 6.3: Bottleneck throughputs with curved-line platooning for scenarios with two incoming lanes, varying the maximum platoon length

The figures show the bottleneck throughputs for varying *max platoon lengths* for respectively the scenarios 'Crossing-turn', 'T-merge', and 'Ramp' with curved-line platooning. All three figures show a higher *max platoon length* corresponds to a higher bottleneck throughput, with the 'One lane blocked' graph functioning as an asymptote.

Straight-line platooning improves the bottleneck throughputs for the scenarios 'Straight', 'Crossing', and 'Loading'. In the scenario 'Ramp' the average bottleneck throughput increases, but the throughput of the corner lane can decrease as a result of straight-line platooning. Curved-line platooning improves the bottleneck throughput for all scenarios with respect to the *AgvSorter* baseline. For scenarios with multiple incoming lanes a balance should be found in optimizing bottleneck throughput without delaying individual AGVs too long. This is influenced by the coordination rules. Because of the improved bottleneck throughputs, the platooning strategies are expected to substantially increase system throughput, especially curved-line platooning with lenient cutting corners and platoon coordination.

6.2 Benchmark system configurations simulation results

This section analyzes the effect of straight-line platooning and the coordination rules on the system throughput and lead time percentile by means of simulation. Additionally, it is analyzed how the indication delay parameter affects the system performance. First, the design of the simulation experiments is discussed, and then the simulation experiment results are presented and analyzed.

6.2.1 Design of the simulation experiments

To investigate the effects of straight-line platooning and the coordination rules, simulations are performed for the following cases:

- Unrestricted straight-line platooning
- Straight-line platooning with a permitted platoon length strategy, with varying maximum platoon lengths of 2, 3, 5, 8, and 12 AGVs
- Straight-line platooning with a RoW true platoon length strategy, with varying maximum RoW platoon lengths of 2, 3, 5, 8, and 12 AGVs
- Straight-line platooning with a RoW observed platoon length strategy, with varying maximum RoW platoon lengths of 2, 3, 5, 8, and 12 AGVs

A value for the maximum platoon length of 12 restricts only long platoons, 5 and 8 additionally restrict medium-long platoons, and 2 and 3 also restrict short platoons. As the *AgvSorter* should be applicable to different system layouts and AGV densities, simulations are performed for all three benchmark system configurations, as defined in Chapter 3, with densities of 20% and 25%. By default, a value for the indication delay of 0.2 s is used. This value results in a small inter-vehicular distance, for which it seems still achievable to ensure safety in an actual physical version of the system. reasonably Throughout the simulations, all other model configuration options are kept constant.

In separate simulations, the influence of the indication delay is also investigated. Simulations are ran on the *GridSorter* configuration with a density of 20% for the following values of the indication delay parameter: 0, 0.2, 0.4, 0.6, 0.8 s.

Often simulation studies are conducted by single simulations of arbitrary length, and treating the simulation outputs as the true behavior of the simulated model. However, as random samples from probability distributions are often used to progress through time, these simulation outputs are just a single possible realization, and can largely differ from the true model behavior. A simulation should be considered as a computer-based statistical sampling experiment, hence apt statistical techniques should be applied in designing and analyzing the simulation experiments. [27]

Assume a sampling population of N simulations, all with an identical system configuration, with a single outcome variable, e.g. system throughput. Then, the sample mean μ^* of the outcome is an unbiased estimator of the actual mean μ. The sample mean is determined according to Equation 6.1

$$\mu^* = \frac{1}{N} \sum_{i=1}^{N} X_i \tag{6.1}$$

with X_i being the outcome of simulation sample i. The standard error S is an unbiased estimator of the variance of μ^*, and can be determined according to Equation 6.2. S can be interpreted as an indicator for the deviation between the sample mean and the actual mean.

$$S = \sqrt{\frac{1}{N(N-1)} \sum_{i=1}^{N} [X_i - \mu^*]^2} \tag{6.2}$$

For $N \to \inf$, $\mu^* \to \mu$ and $S \to 0$. More simulations give a better estimate of system behavior. The standard error can be used to determine the confidence interval for μ, assuming the simulation outcome samples are normally distributed, by means of the student-t distribution [27]. The $100(1-\alpha)$ confidence interval is given by Equation 6.3

$$\mu^* \pm t_{N-1,1-\alpha/2} S \tag{6.3}$$

with $t_{N-1,1-\alpha/2}$ being the upper $1-\alpha/2$ critical point for the t distribution with $n-1$ degrees of freedom. Because of the central limit theory, for sufficiently large values of N, the simulation samples are distributed according to a normal distribution. A sample population of 30 simulations is assumed sufficient [28]. Therefore, for every system configuration considered in the analysis, 30 simulations are performed.

Additionally, it is also important to smartly choose the length of a single simulation. A simulation consists of a series of processes, such as loading, and turning. The summation of all these processes results in the system behavior. For short simulation times, single processes can have a relatively big effect on the simulation outcome, resulting in a biased system behavior estimate. Ideally, to mitigate this effect, simulation times should go to infinity. This is computationally too expensive. It is chosen to use a simulation time of 1200 s for several reasons: (1) underlying simulation processes take only a minor part of 1200 s, e.g. 11000 platoon actions in GridSorter2 with a 20% density with unrestricted straight-line platooning; (2) it is acceptable in terms of computation time; (3) previous projects regarding the *AgvSorter* also used this simulation time.

6.2.2 Analysis of the simulation experiments

In analyzing different system configurations, it is important to be able to tell when one system configuration performs better than another. Showing a significant difference between the performance of two system configurations, can be done using a paired-t $100(1-\alpha)$ confidence interval, with $\alpha = 0.05$. This is possible because for every analyzed system configuration exactly 30 simulations are performed. With this confidence interval,

it can be verified if the null hypotheses $H_0 : \mu_1 = \mu_2$ can be rejected against $H_1 : \mu_1 \neq \mu_2$, with μ_1 being the expected outcome from configuration 1, and μ_2 being the expected outcome from configuration 2. For this analysis a 0.05 significance is used. [27]

According to [29], H_0 can be rejected when the overlap of the 95% unpaired confidence intervals is less or equal to one quarter of the average length of the two confidence intervals, if sample sizes for both configuration are bigger than 10, and the interval lengths do not differ a factor two.

Results system throughput

The main KPI for the *AgvSorter* is the system throughput. The system throughputs for the *AgvSorter* with the different strategies are determined using simulations. The throughput results of the simulations are presented next, according to Table 6.2. All these results are normalized for the sample mean throughput of the baseline *AgvSorter* model with a pivoting strategy. In these figures, the horizontal axis shows the maximum platoon length.

Figure	**Layout**	**Density**	**Turning strategy**
6.4	*GridSorter*	20%	Pivoting
6.5	*GridSorter*	25%	Pivoting
6.6	*GridSorter*	20%	Cutting corners
6.7	*GridSorter*	25%	Cutting corners
6.8	*Airport*	20%	Pivoting
6.9	*Airport*	25%	Pivoting
6.10	*Airport*	20%	Cutting corners
6.11	*Airport*	25%	Cutting corners
6.12	*Tessellation*	20%	Pivoting
6.13	*Tessellation*	25%	Pivoting
6.14	*Tessellation*	20%	Cutting corners
6.15	*Tessellation*	25%	Cutting corners

Table 6.2: Overview results for system throughput

GridSorter

Next, the system throughput results are presented for the *GridSorter* configurations.

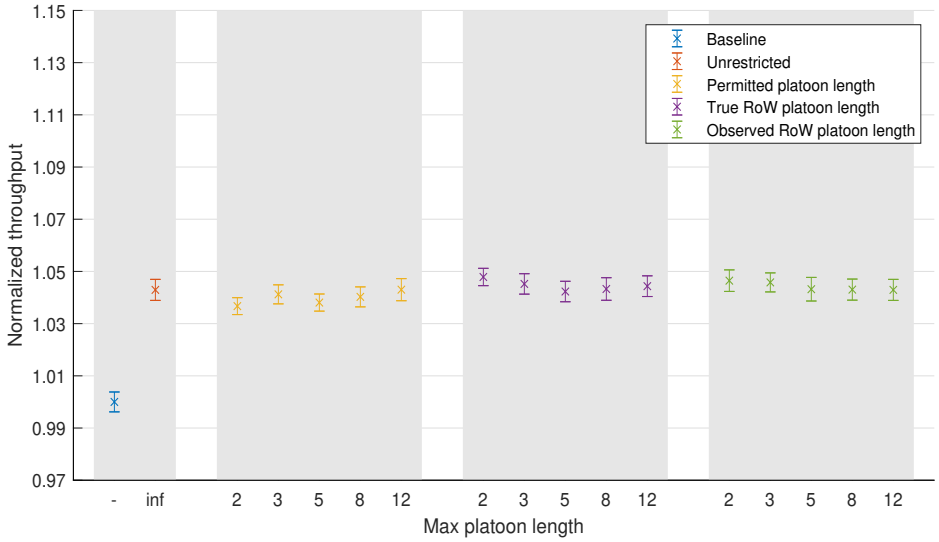


Figure 6.4: Results system throughput for the *GridSorter* layout, with an AGV density of 20%, with pivoting

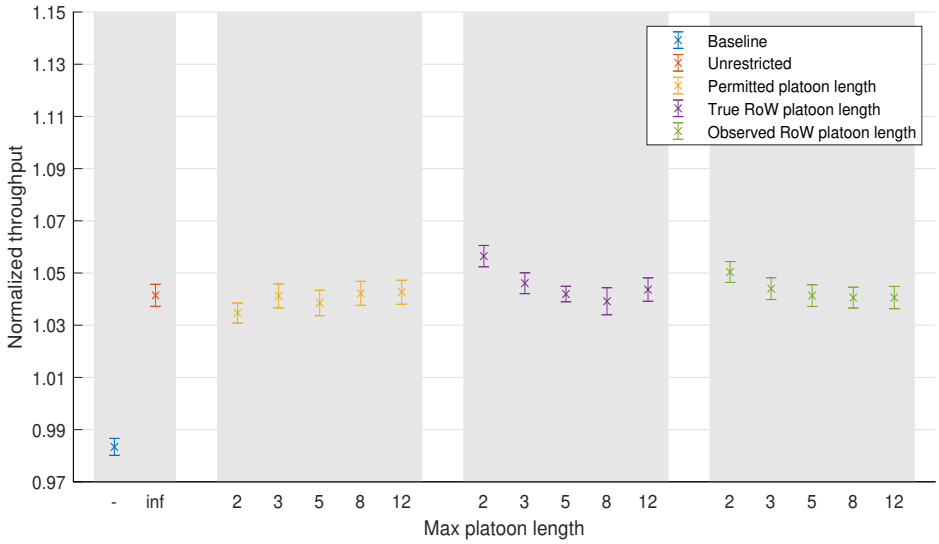


Figure 6.5: Results system throughput for the *GridSorter* layout, with an AGV density of 25%, with pivoting

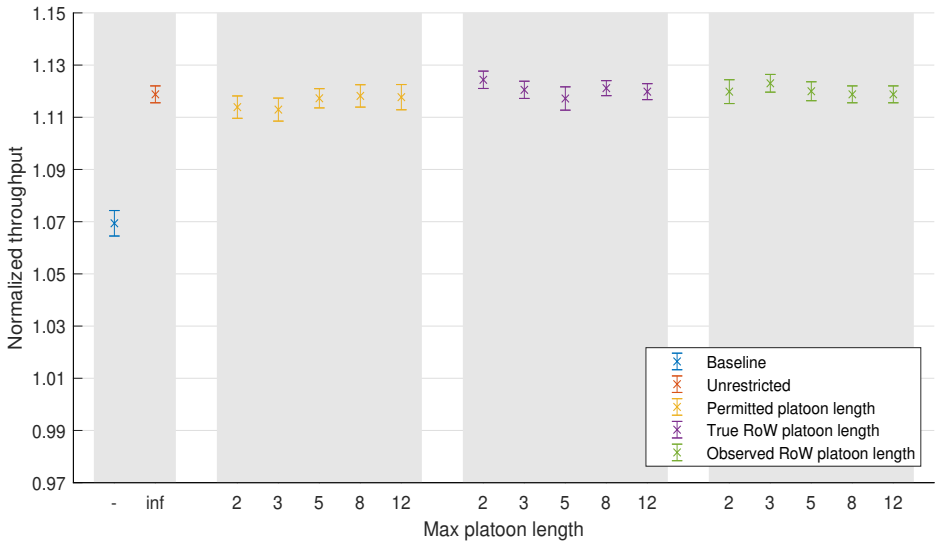


Figure 6.6: Results system throughput for the *GridSorter* layout, with an AGV density of 20%, with cutting corners

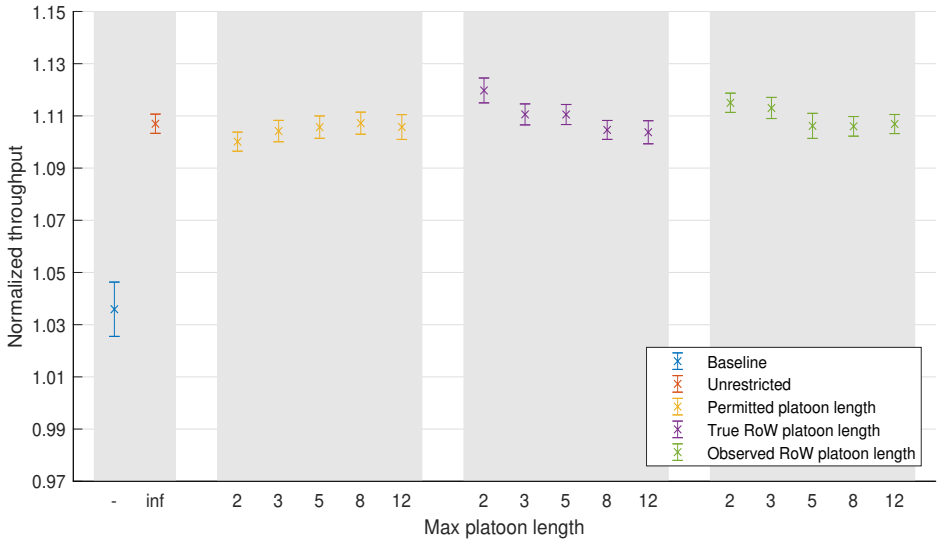


Figure 6.7: Results system throughput for the *GridSorter* layout, with an AGV density of 25%, with cutting corners

Airport

Next, the system throughput results are presented for the *Airport* configurations.

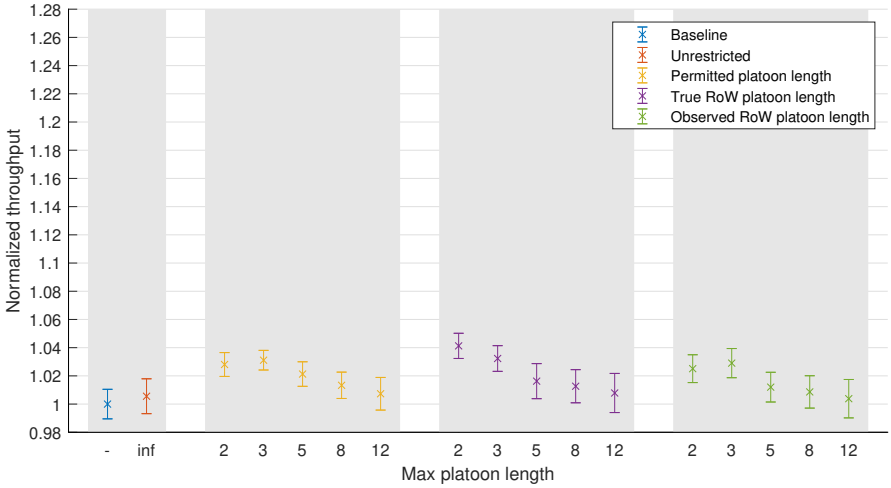


Figure 6.8: Results system throughput for the *Airport* layout, with an AGV density of 20%, with pivoting

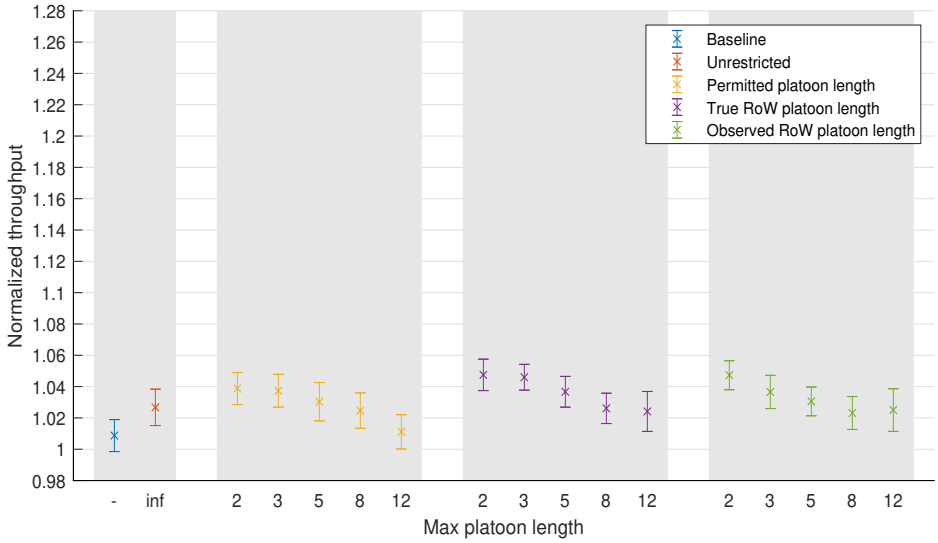


Figure 6.9: Results system throughput for the *Airport* layout, with an AGV density of 25%, with pivoting

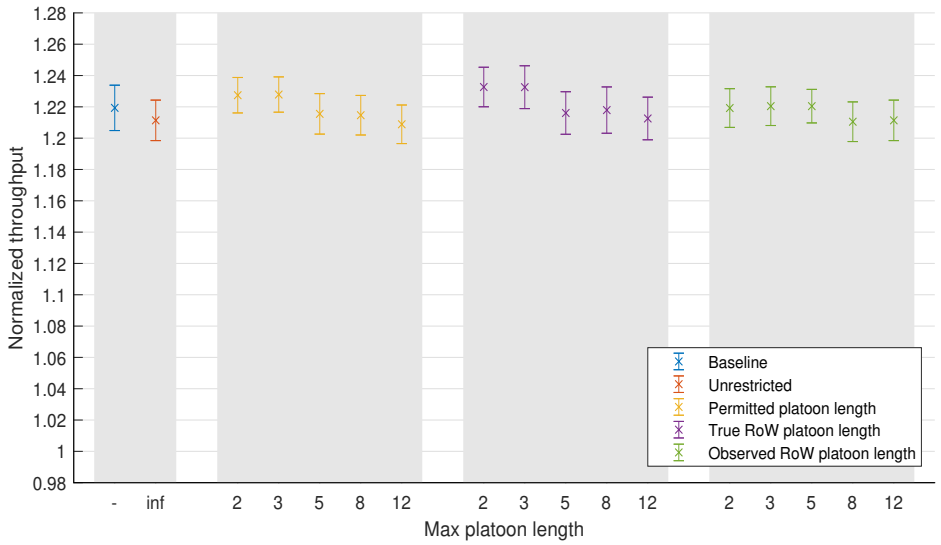


Figure 6.10: Results system throughput for the *Airport* layout, with an AGV density of 20%, with cutting corners

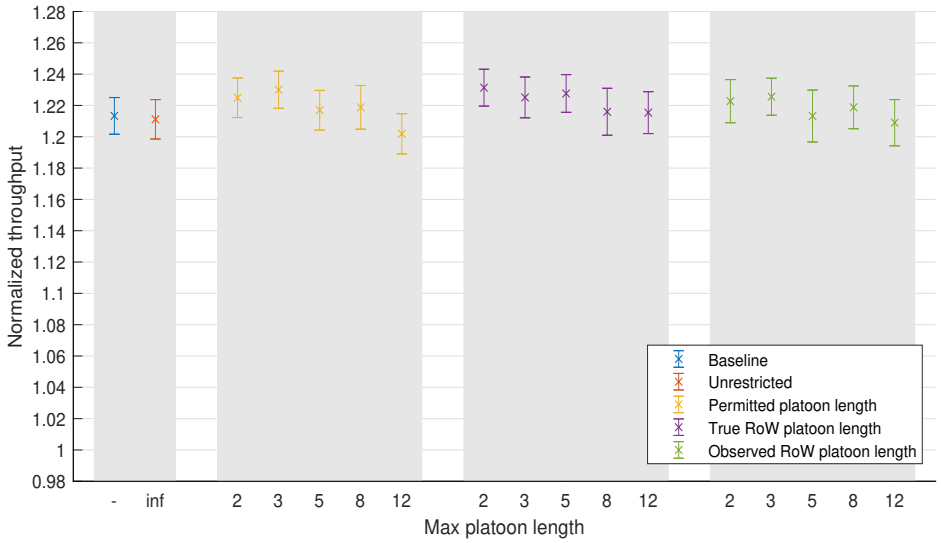


Figure 6.11: Results system throughput for the *Airport* layout, with an AGV density of 25%, with cutting corners

Tessellation

Next, the system throughput results are presented for the *Tessellation* configurations.

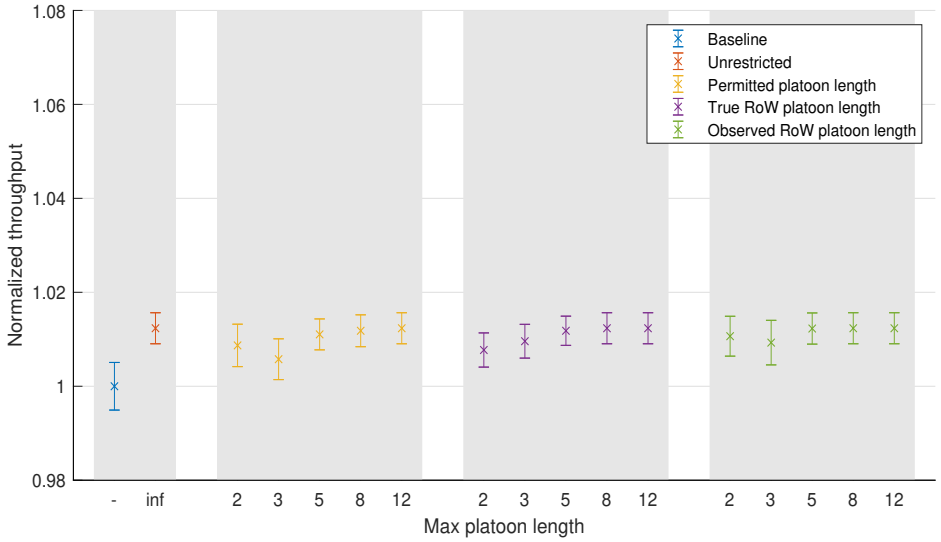


Figure 6.12: Results system throughput for the *Tessellation* layout, with an AGV density of 20%, with pivoting

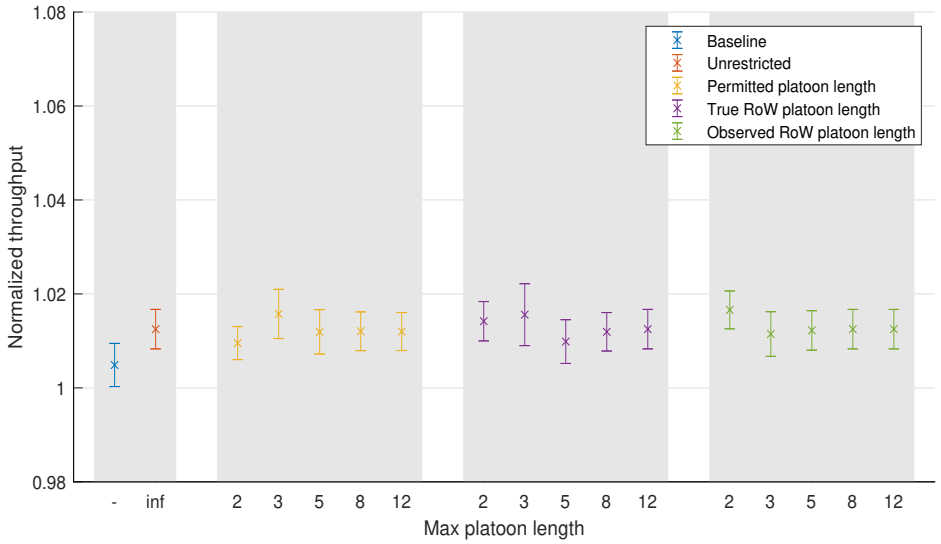


Figure 6.13: Results system throughput for the *Tessellation* layout, with an AGV density of 25%, with pivoting

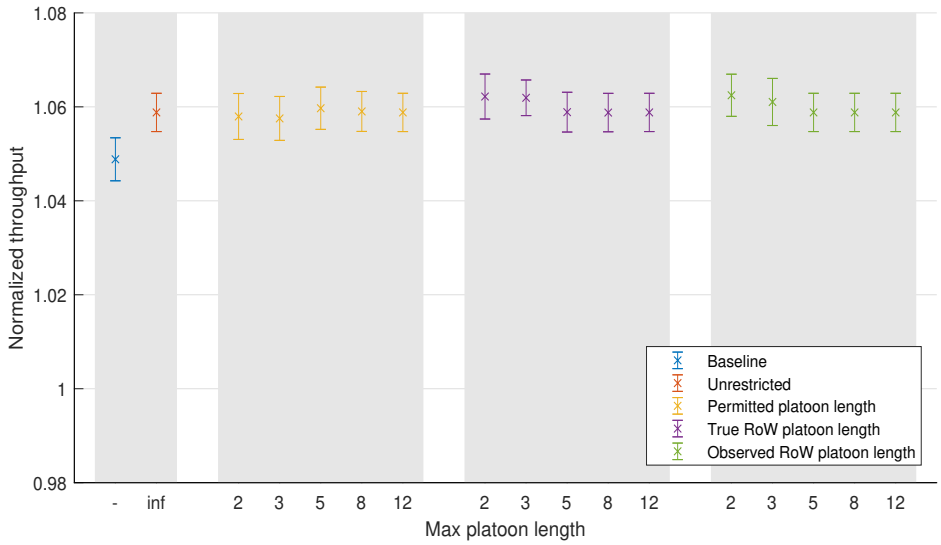


Figure 6.14: Results system throughput for the *Tessellation* layout, with an AGV density of 20%, with cutting corners

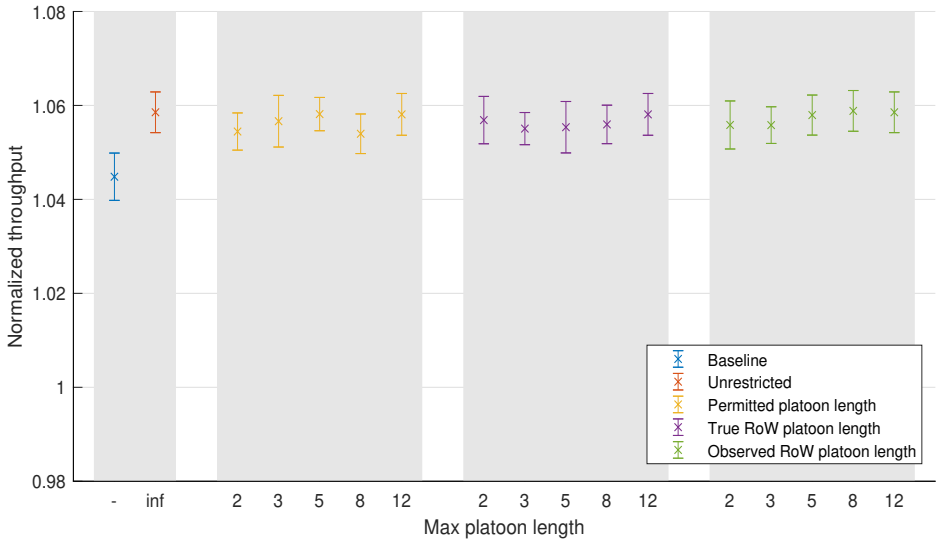


Figure 6.15: Results system throughput for the *Tessellation* layout, with an AGV density of 25%, with cutting corners

Discussion system throughput

For all four *GridSorter* configurations, straight-line platooning results in a 4% to 5% throughput improvement with respect to the baseline. The coordination rules do not seem to affect the system throughput for a density of 20%. From the visualization of the simulation model, it is obtained that at this density, platoons longer than five only seldom occur, and most platoons are of length two or three. Therefore, the coordination rules have relatively little effect. At a density of 25%, the effect of the coordination rules becomes apparent. The permitted platoon length strategy does not seem to significantly affect the system throughput. However, the RoW platoon length strategies with short max platoon lengths (2 or 3) significantly improve system performance with respect to unrestricted platooning. For the RoW platoon length strategies the true RoW platoon length strategy has advantage over the observed RoW platoon length strategy.

For the *Airport* configurations, unrestricted straight-line platooning does not significantly change system throughput with respect to the baseline. This corresponds to expectations, as the throughput of segment-type layout configurations was hypothesized to be greatly dependent on the benchmark scenarios with low bottleneck throughputs. Additionally, a side effect of unrestricted platooning can be for AGVs to have to wait for long platoons to pass, before being able to continue. In grid-type layouts the effect of this side effect is mitigated by the possibility of rerouting. However, in segment-type layouts, rerouting is often not possible. This can substantially delay groups of AGVs, negatively affecting the system throughput. The coordination rules are designed to reduce this effect, and as expected, the coordination rules tend to improve the system throughput. The permitted platoon length strategy seems to positively affect the system throughput for short max platoon lengths, but with the available data only for a 20% density with a pivoting strategy, a significant difference is established. A similar effect is shown in the figures for the RoW platoon length strategies. For these strategies, with the available data, a significant improvement is observed for both AGV densities for a pivoting strategy, and also for a 20% density with a cutting corners strategy. The maximum throughput improvement achieved by the strategies with respect to the baseline is approximately 4% for a pivoting strategy, and no significant improvement for a cutting corners strategy.

For the *Tessellation* configurations, straight-line platooning results in a small but significant improvement ($\pm 1\%$). A reason for the improvement being relatively small, especially compared to the *GridSorter* configurations, can be found in the long total segment length of the *Tessellation* layout in comparison to the total segment length of the *GridSorter* layout. The same amount of AGVs drives distributed over a longer total segment length. As a result, less platooning actions are performed, and platooning has a smaller effect on system throughput. Additionally, from analysis using the visualization of the simulation model, it is found that AGVs in *Tessellation* can relatively often not reserve a tile to prevent the occurrence of a deadlock. For this reason, it is often necessary to interrupt platooning actions, again reducing the effect of platooning on system throughput. As a result, long platoons do not occur frequently in hexagonal layouts. This is also an explanation for the coordination rules not having a notable effect on system throughput, as coordination rules only act in situations with platoons longer than the max platoon length.

Based on the *Gridsorter* results, the throughputs of grid-type layouts seem to substantially improve as a result of straight-line platooning. Based on the *Airport* results, throughputs of segment-type configurations seem to only improve with platoon coordination, as platoon coordination mitigates the side effects of unrestricted platooning. Based on the *Tessellation* results, hexagonal configurations seem to significantly, but not substantially improve from straight line platooning both with and without platoon coordination.

Results lead time percentile

Another KPI for the *AgvSorter* is the lead time percentile. The lead time percentiles for the *AgvSorter* with the different strategies are determined using simulations. The lead time percentile results are presented next, according to Table 6.3. All these results are normalized for the sample mean lead time percentile of the baseline *AgvSorter* model with a pivoting strategy. In these figures, the horizontal axis shows the maximum platoon length.

Figure	Layout	Density	Turning strategy
6.16	*GridSorter*	20%	Pivoting
6.17	*GridSorter*	25%	Pivoting
6.18	*GridSorter*	20%	Cutting corners
6.19	*GridSorter*	25%	Cutting corners
6.20	*Airport*	20%	Pivoting
6.21	*Airport*	25%	Pivoting
6.22	*Airport*	20%	Cutting corners
6.23	*Airport*	25%	Cutting corners
6.24	*Tessellation*	20%	Pivoting
6.25	*Tessellation*	25%	Pivoting
6.26	*Tessellation*	20%	Cutting corners
6.27	*Tessellation*	25%	Cutting corners

Table 6.3: Overview results for lead time percentile

GridSorter

Next, the lead time percentile results are presented for the *GridSorter* configurations.

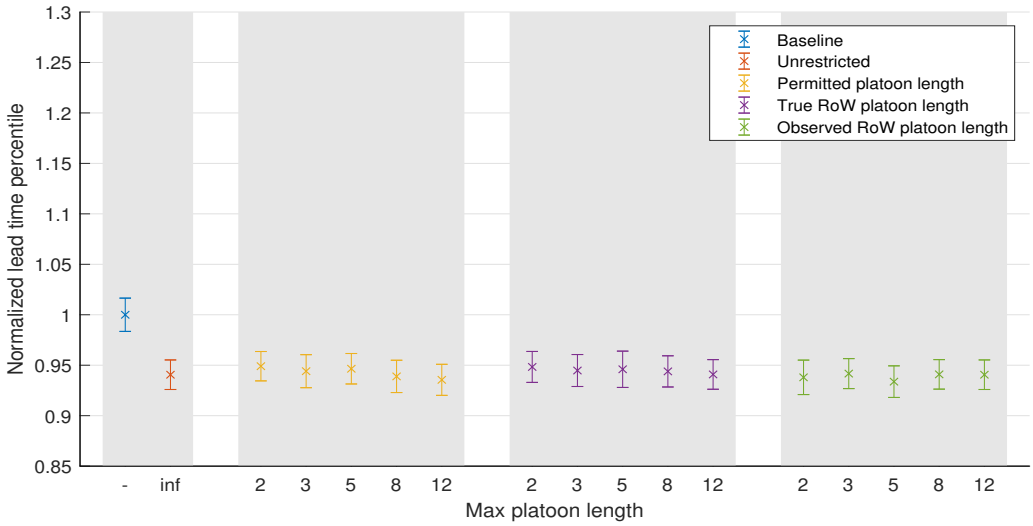


Figure 6.16: Results lead time percentile for the *GridSorter* layout, with an AGV density of 20%, with pivoting

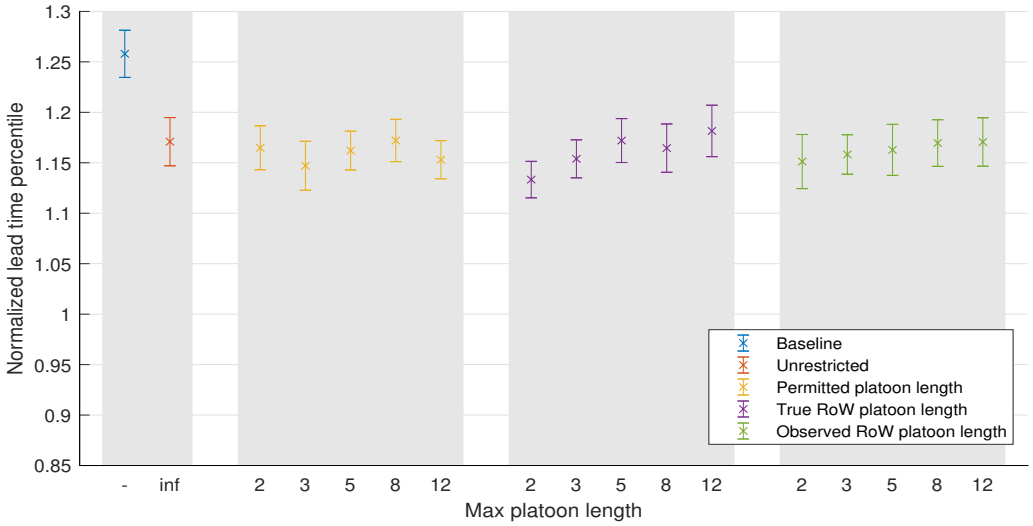


Figure 6.17: Results lead time percentile for the *GridSorter* layout, with an AGV density of 25%, with pivoting

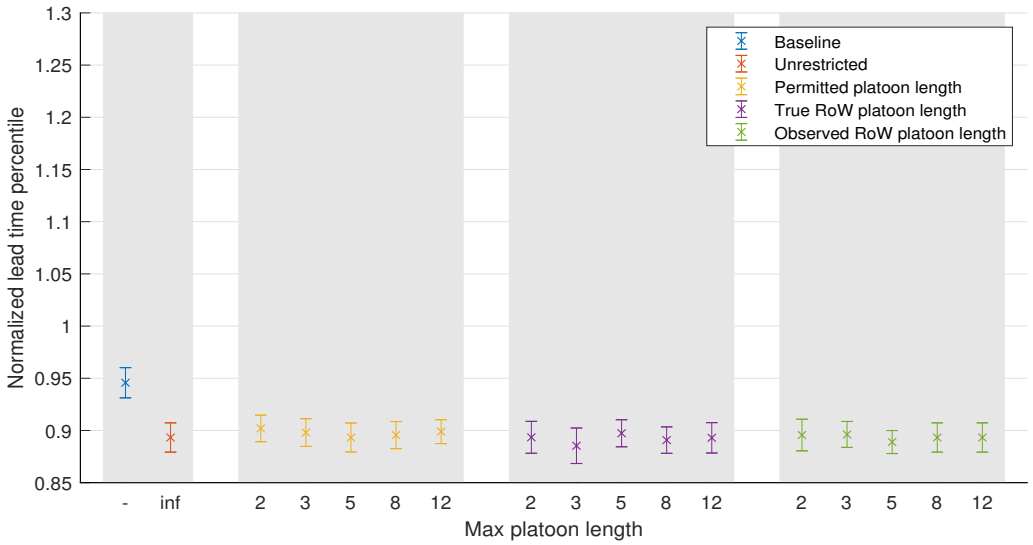


Figure 6.18: Results lead time percentile for the *GridSorter* layout, with an AGV density of 20%, with cutting corners

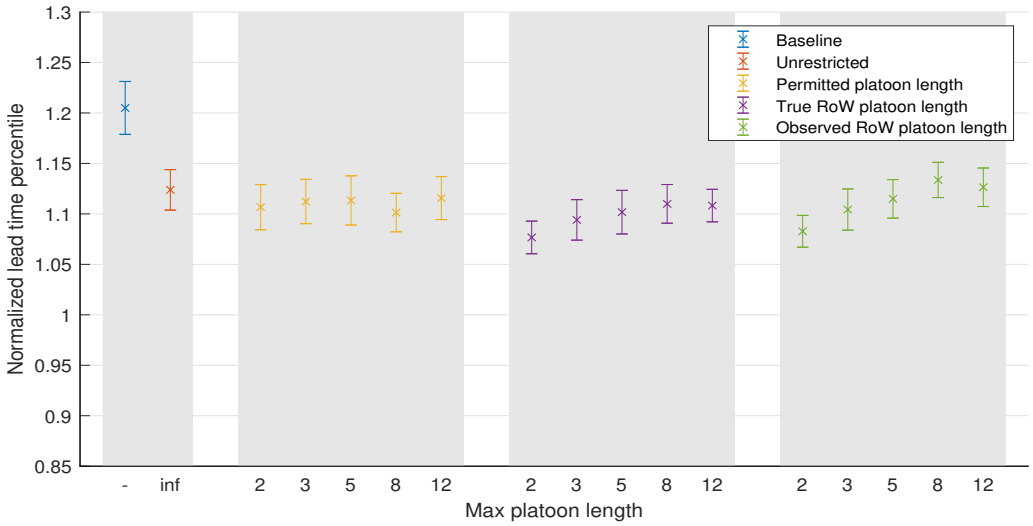


Figure 6.19: Results lead time percentile for the *GridSorter* layout, with an AGV density of 25%, with cutting corners

Airport

Next, the lead time percentile results are presented for the *Airport* configurations.

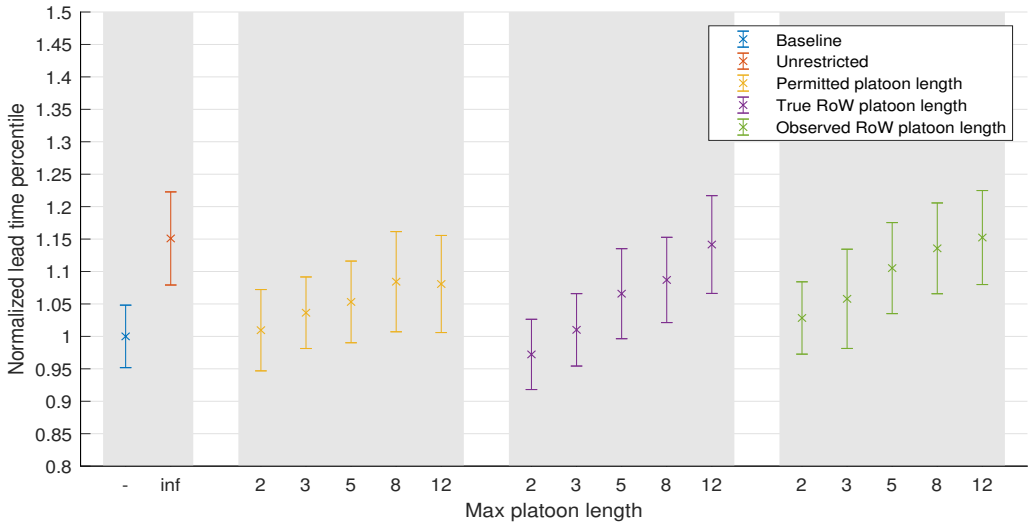


Figure 6.20: Results lead time percentile for the *Airport* layout, with an AGV density of 20%, with pivoting

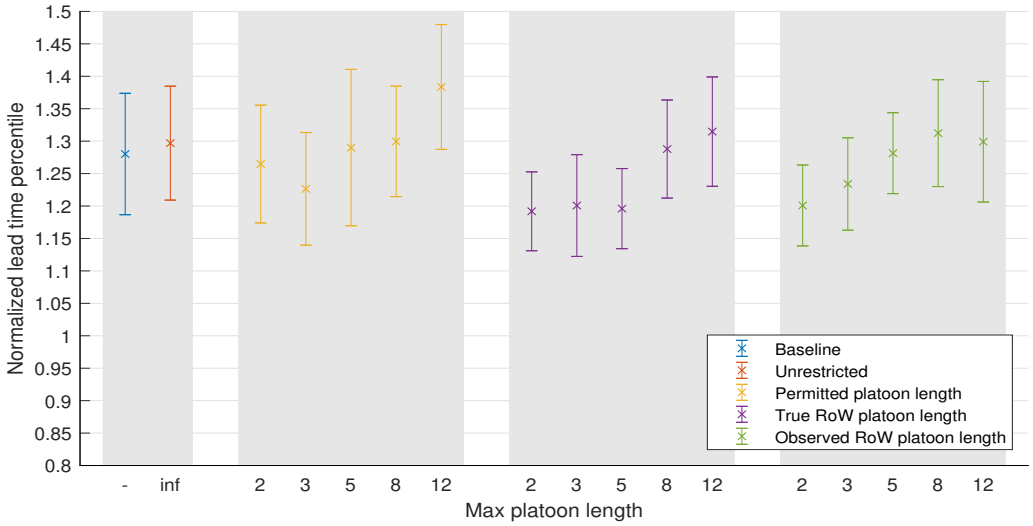


Figure 6.21: Results lead time percentile for the *Airport* layout, with an AGV density of 25%, with pivoting

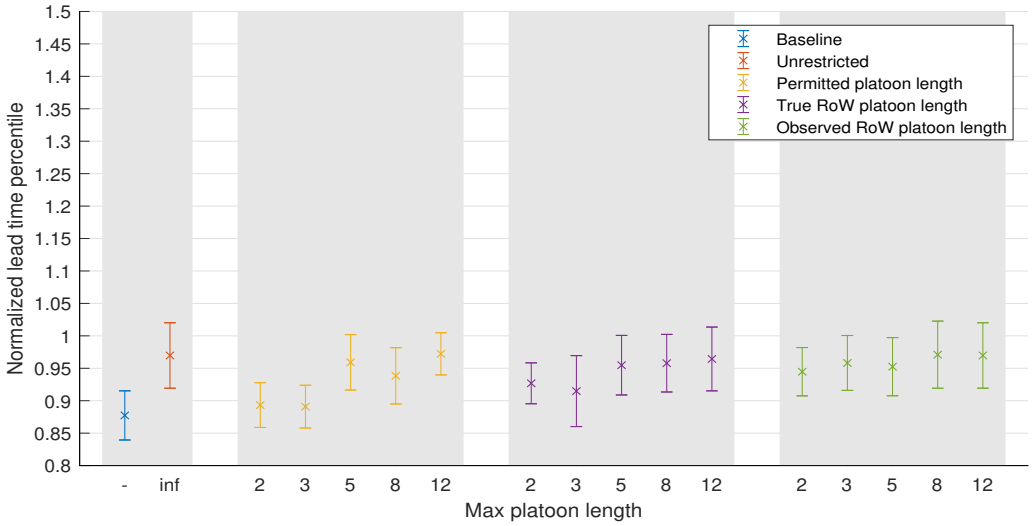


Figure 6.22: Results lead time percentile for the *Airport* layout, with an AGV density of 20%, with cutting corners

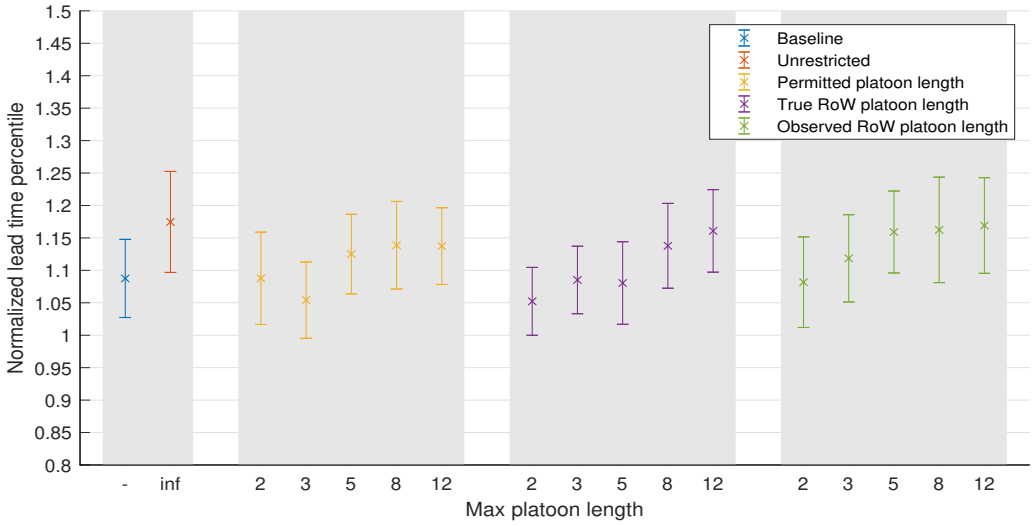


Figure 6.23: Results lead time percentile for the *Airport* layout, with an AGV density of 25%, with cutting corners

Tessellation

Next, the lead time percentile results are presented for the *Tessellation* configurations.

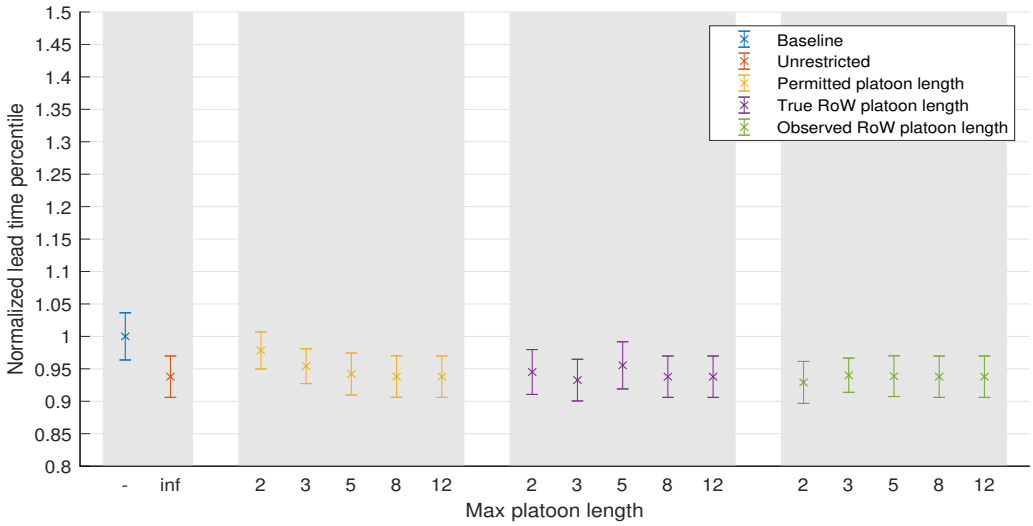


Figure 6.24: Results lead time percentile for the *Tessellation* layout, with an AGV density of 20%, with pivoting

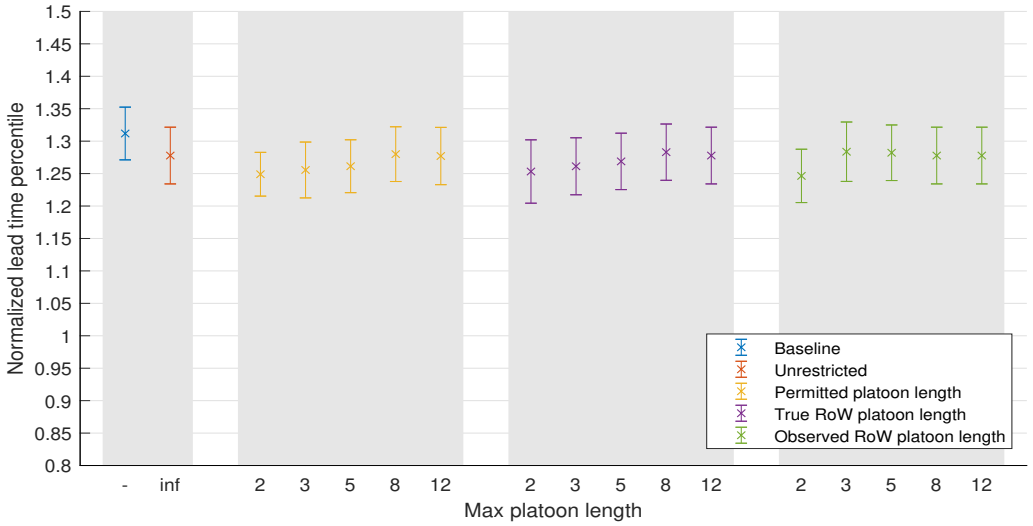


Figure 6.25: Results lead time percentile for the *Tessellation* layout, with an AGV density of 25%, with pivoting

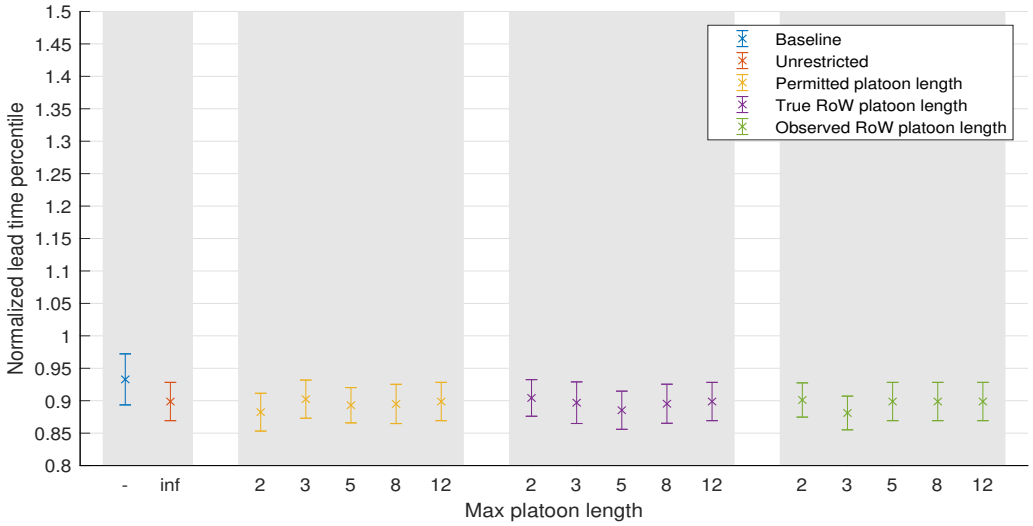


Figure 6.26: Results lead time percentile for the *Tessellation* layout, with an AGV density of 20%, with cutting corners

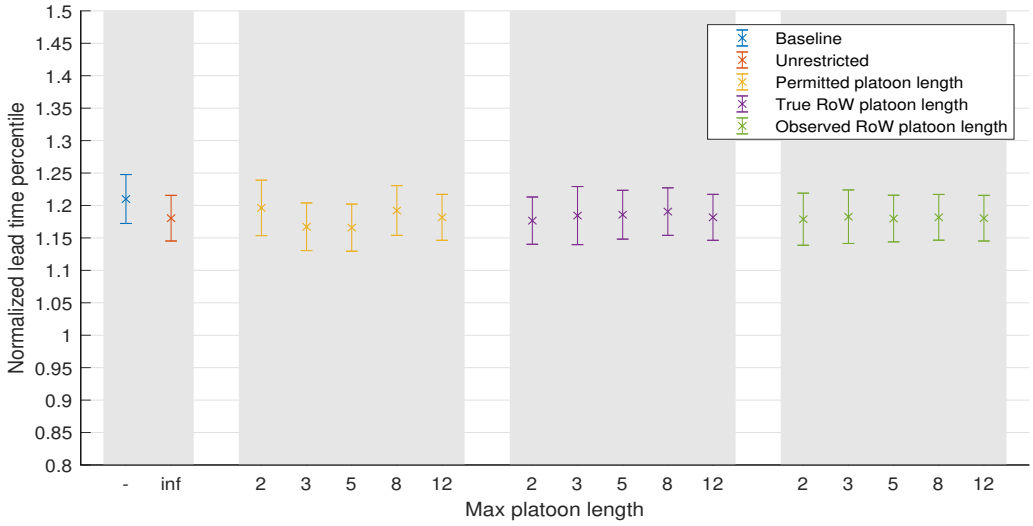


Figure 6.27: Results lead time percentile for the *Tessellation* layout, with an AGV density of 25%, with cutting corners

Discussion lead time percentile

It is undesirable for any item to remain in the system disproportionately long. Therefore, it is desired to not have a high lead time percentile. For all four *GridSorter* configurations, straight-line platooning improves the lead time percentile with approximately 5%. For a 20% density, the coordination rules do not show a substantial effect, as for this density long platoons do only rarely occur. For a 25% density, lead time percentile is especially reduced by the true RoW platoon length strategy with a low max platoon length (2 or 3). Naturally, higher densities have higher job lead times, as there is more traffic in the system.

For the *Airport* configurations, unrestricted straight-line platooning negatively affects the lead time percentile. This is likely caused by the before mentioned side effect that AGVs have to wait for indefinitely long for a platoon to pass, before being able to continue. This effect is mitigated by the coordination rules. Especially a true RoW platoon length strategy with low max platoon lengths cancels out this effect, but permitted platoon length and observed RoW platoon have a similar positive influence.

For the *Tessellation* configurations with a 20% density, straight-line platooning significantly reduces the lead time percentile. The results show no effect from applying different coordination strategies. For a 25% density, no conclusions can be drawn with respect to the baseline. Nevertheless, it can be concluded that coordination with low max platoon lengths results in a lower lead time percentile than unrestricted straight-line platooning.

In evaluating the lead time percentile, it is mostly important that this value is not disproportionately high with respect to the baseline, as it is important that no item remains in the system too long. The results show platooning results in lower or approximately equal values for the lead time percentile, indicating items do not for any type of configuration remain in the system disproportionately long.

6.2.3 The effect of the indication delay parameter

In previously described simulations, the effect of the indication delay parameter has been neglected. A standard value of 0.2 s has been used. However, it was hypothesised that longer delays would negatively affect system throughput. In Figure 6.28, the system throughput for the *GridSorter* configuration with an AGV density of 20%, a pivoting strategy and unrestricted straight-line platooning is shown for different values of the indication delay parameter. To put these times in perspective, for the *GridSorter* configuration, an AGV takes 1.3 s to clear a tile from standstill. Shown in the figure are the sample mean throughput and the 95% confidence interval around it, normalized for the throughput of the baseline *AgvSorter* model.

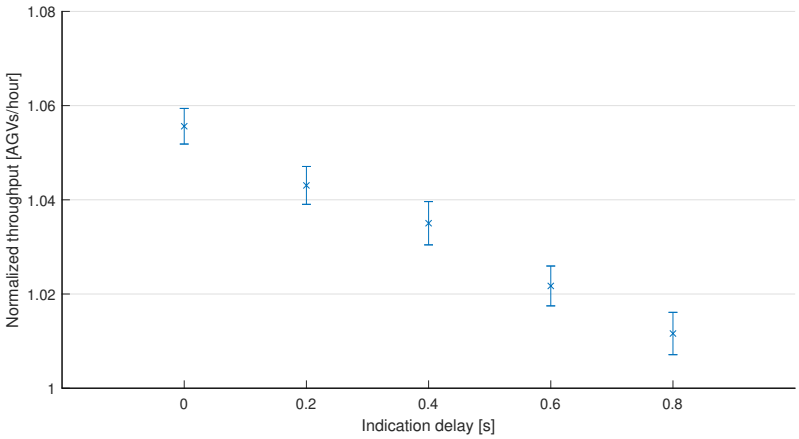


Figure 6.28: The influence of the indication delay parameter on the throughput

As can be seen in the figure, similar to expectations, a longer delay results in a lower system throughput. The results seem to suggest an almost linear relationship between the indication delay and the system throughput. Therefore, in order to optimize the effect of platooning, communication with the AGVs should be as quick and frequent as possible.

6.3 The effect of grid-based platooning on the system's quality attributes

The four quality attributes that are especially relevant for the *AgvSorter* system are: safety, scalability, flexibility, and robustness.

- *Safety:* AGVs in a platoon have a relatively small inter-vehicular distance. This increases the risk of collisions. To ensure safety, AGVs must be able to react to sudden stops of preceding vehicles more quickly. This can be achieved by good AGV hardware, such as accurate on-board distance sensors and strong brakes. More research for real-world systems is required to determine the minimum inter-vehicular distance at a given speed for given AGV hardware.

- *Scalability:* It is still possible to increase system performance by increasing system resources. Performing grid-based platooning does require higher computational and memory resources. It is necessary for the system to operate real time, and more operations per time unit have to be executed. Also, more values need to be temporarily stored in memory. Similar to the baseline *AgvSorter*, it is nonsensical to indefinitely add AGVs to the system. Every system configuration has an optimal AGV density. For the baseline *AgvSorter*, this optimal AGV density tends to be in between 20% and 25%. It is expected that grid-based platooning increases this optimum density, because AGVs in a platoon occupy less space in the system per AGV. This would make it possible to increase system performance by adding AGVs up to a higher AGV density. To be able to make this claim, more research is necessary.

- *Flexibility:* Grid-based platooning does not affect the flexibility of an AGV system.
- *Robustness:* In the baseline *AgvSorter* system, if an AGV blocks one or two tiles as a result of an AGV malfunctions, it is often possible for AGVs to circumvent this blockage by rerouting. For grid-based platooning, if a leader AGV malfunctions, this can block multiple follower vehicles in the platoon. Rerouting is only possible from the center of a tile. If a different route is available, in the baseline *AgvSorter*, AGVs can always reroute. However, when applying a grid-based platooning strategy, if a follower of a blocked AGV has already passed the tile center, it is also blocked, causing a chain effect, see Figure 6.29. In the example red AGVs are blocked, and green AGVs are non-blocked. In reality, chances of such long chains of blocked AGVs to occur are small, and only zero to two extra AGVs are affected, making the effect almost negligible.

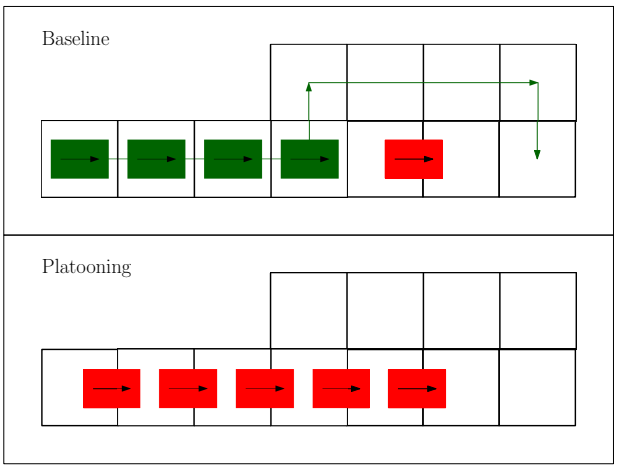


Figure 6.29: The effect of a single AGV malfunction for the baseline *AgvSorter*, and for the *AgvSorter* with grid-based platooning

6.4 Summary

This chapter discussed how grid-based platooning influences the *AgvSorter* system. Grid-based platooning increases the bottleneck throughput for several benchmark scenarios. Curved-line platooning even substantially improved the bottleneck throughputs for all benchmark scenarios. Therefore, curved-line platooning is expected to substantially increase system throughput. The effect of straight-line platooning with different coordination rules is also analyzed by performing simulations on the benchmark system configurations. Especially for grid-type layouts increased system throughputs are achieved. Also for segment and hexagonal layouts a small but significant improvement in system throughput is achieved. Additionally, the lead time percentile results suggest items do not for any type of configuration remain in the system disproportionally long. The effect of the indication delay parameter on throughput was investigated, and as expected a shorter delay results in a higher system throughput. Finally, the effect of grid-based platooning on the system's quality attributes was investigated. More research is required towards determining the effect of grid-based platooning on safety and scalability; flexibility is not affected, and a small almost negligible compromise is made on robustness.

7: Conclusions & recommendations

This thesis describes a study on the design of a grid-based platooning strategy for the dense *AgvSorter*, and analyzes its effects on performance. The main KPI for the *AgvSorter* is the system throughput, but also the lead time percentile and the bottleneck throughputs are system KPIs. Additionally, four quality attributes are important for the *AgvSorter*: safety, scalability, flexibility, and robustness. The goal of designing a grid-based platooning strategy is to improve KPIs, without compromising on quality attributes. In the first section of this chapter, the conclusions for this thesis are described. In the second section, recommendations are made for future research regarding grid-based platooning and for the application of grid-based platooning.

7.1 Conclusions

To be able to design a grid-based platooning strategy, research is conducted towards existing platooning systems and technology. It was found that platooning research can be subdivided into platoon control, dealing with intra platoon interactions, and platoon coordination, dealing with inter platoon interactions. In platoon control, important tasks are ensuring collision-free driving, minimizing inter-vehicular distance, and guaranteeing string stability. However, to the best of knowledge, no platoon control research applicable to grid-based systems exists in literature. In platoon coordination, important tasks include regulating manoeuvres and regulating platoons and vehicles efficiently through bottlenecks in the system. For a grid-based platooning strategy, such coordination strategies might improve system performance, and can be used as inspiration in the design process.

Before designing a grid-based platooning strategy, the baseline *AgvSorter* was described and analyzed. To analyze the *AgvSorter* system, benchmark scenarios and benchmark configurations are constructed. From the baseline *AgvSorter* analysis it was hypothesized that platooning could bring a gain in system performance.

First a strategy is designed for AGVs driving in a straight line: straight-line platooning. It is based on the leader-follower principle. The strategy ensures safety and optimal behavior. However, it does not improve the scenarios with low bottleneck throughputs, such as ‘T-merge’ and ‘Curve’. Therefore, a platooning strategy is designed allowing for AGVs to cut a corner as a platoon: curved-line platooning, aiming to improve low bottleneck throughputs. It also works according to the leader-follower principle. Curved-line platooning shares the disadvantage with straight-line platooning that AGVs outside of

a platoon have to wait indefinitely long for a platoon to pass, before being allowed to continue. Curved-line platooning only occurs if a leader vehicle decides to cut a corner. However, the baseline cutting corners is conservative in deciding to cut a corner or not. Therefore, a more lenient cutting corners strategy is designed, that allows AGVs to cut a corner in more situations. To solve that AGVs outside of a platoon have to wait indefinitely long for platoons to pass, three coordination strategies preventing this behavior are designed: permitted platoon length, RoW true platoon length and RoW observed platoon length. All three strategies prevent AGVs to indefinitely have to wait for platoons to pass. It is difficult to say which strategy performs best without performing simulations.

Straight-line platooning and the three coordination rules are implemented in a MATLAB simulation model. To check correctness of the implementation, various validation and verification steps are performed.

To analyze the performance of the designed grid-based platooning strategies, the system is evaluated on its KPIs using analyses and simulations. The bottleneck throughputs for the benchmark scenarios are calculated for the different strategies, and used to compare the different strategies to the baseline *AgvSorter*. Straight-line platooning improves the bottleneck throughputs for several scenarios. Curved-line platooning with lenient cutting corners improves the bottleneck throughputs for all benchmark scenarios. To investigate the effects of the implemented strategies on the system throughput and lead time percentile, simulations are performed using the benchmark configurations. Especially for grid-type layouts an increased system throughput is achieved. For segment and hexagonal layouts a small but significant improvement in system throughput is achieved. The results show a beneficial effect from platoon coordination, especially for segment type layouts coordination is necessary to prevent AGVs having to wait for long platoons to cross. Additionally, the results regarding the lead time percentile suggest items do not remain disproportionally long in the system for any layout type. The effect of the indication delay parameter is investigated, and as expected, a shorter delay results in better system performance. Finally the effect of grid-based platooning on the system's quality attributes is investigated. More research is necessary towards determining the effects on safety and scalability; flexibility is not affected, and a small almost negligible compromise is made on robustness.

As curved-line platooning and lenient cutting corners are not implemented in the MATLAB simulation model, no simulations are performed for these strategies. Grid-type layouts are hypothesized to benefit from improved bottleneck throughput for any scenario. Segment-type layouts mostly benefit from improving scenarios with low bottleneck throughputs. Because curved-line platooning improves all bottleneck throughputs for the benchmark scenarios, a substantial improvement in system throughput is expected for both grid-type and segment-type layouts. Hexagonal layouts do, from the obtained results, not show much potential to substantially benefit from grid-based platooning.

7.2 Recommendations

It is advised, based on the results of this thesis, to continue research on grid-based platooning for grid-based AGV systems for grid-type and segment-type layouts. Especially the combination of curved-line platooning with lenient cutting corners with a true RoW platoon length coordination strategy with short maximum RoW platoon lengths seems promising. However, no simulation experiments have been performed for curved-line platooning and lenient cutting corners, because these strategies are not yet implemented in the MATLAB simulation model. The strategies seem promising from the benchmark scenario analysis. Therefore, it is recommended for future research to analyze the effects of cutting corners and lenient cutting corners with simulation based research.

It is theorized that grid-based platooning increases the optimal AGV density for a given layout, as AGVs in a platoon occupy less space in the system per AGV. The optimal AGV density is relevant to achieve optimal system performance, thus it is recommended to further research this claim.

AGVs in a platoon have a relatively small inter-vehicular distance. This increases the risk of collisions. Good AGV hardware, such as strong brakes and accurate on-board distance sensors, can ensure safety for small inter-vehicular distances at high speeds. Before applying grid-based platooning in real-world systems, it is recommended to research the required minimum inter-vehicular distance at a given speed for given AGV hardware.

Performing simulations on large AGV systems is computationally expensive. Therefore, in this thesis simulations with at maximum approximately 100 AGVs and 400 tiles are used. The designed strategies should also be applicable to larger systems. Since the control of the *AgvSorter* is real time, it is recommended for future research to investigate the effects of grid-based platooning on scalability. It is also recommended to explore methods that allow to share the computational work load of operational transportation control over multiple parallel performing processors.

The *AgvSorter* simulation model is based on some assumptions that in a real life system might not be true. Firstly, the perfect world assumption is made, in which system malfunction never occurs, there is no communication loss between different entities in the system, and the motion control of the AGVs is perfect. Secondly, the identical AGVs assumption is made, in which all AGV are identical. Because of these discrepancies between simulation model and real life systems, additional strategy rules might be necessary to ensure safety and optimality for grid-based platooning. Therefore, it is recommended to investigate the effects of these discrepancies on grid-based platooning.

Bibliography

[1] F. Jovane, Y. Koren, and C. R. Boër, "Present and future of flexible automation: Towards new paradigms," *CIRP Annals*, vol. 52, pp. 543 – 560, June 2003. 1

[2] L. Schulze, S. Behline, and S. Buhrs, "Automated Guided Vehicle Systems: a Driver for Increased Business Performance," in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. II, pp. 19 – 21, 2008. 1

[3] "Amazon robotics - our vision." `https://www.amazonrobotics.com/#/vision`. Accessed: 21-05-2019. 1

[4] "Vanderlande - fleet - future-proofing baggage logistics." `https://www.vanderlande.com/airports/evolutions/fleet`. Accessed: 21-05-2019. 1, 2

[5] "Autostore - system." `https://autostoresystem.com/system/`. Accessed: 21-05-2019. 1

[6] "Watch an army of robots efficiently sorting hundreds of parcels per hour." `https://www.youtube.com/watch?v=jwu9SX3YPSk`. Accessed: 21-05-2019. 1

[7] N. C. W. M. Braspenning, J. M. van de Mortel-Fronczak, and J. E. Rooda, "Model-based integration and testing method to reduce system development effort," *Electronic Notes in Theoretical Computer Science*, vol. 164, pp. 13–28, oct 2006. 2

[8] Q. Jin, G. Wu, K. Boriboonsomsin, M. Barth, and M. Barth, "Platoon-based multi-agent intersection management for connected vehicle," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pp. 1462 – 1467, IEEE, oct 2013. 3, 8

[9] J. Vial, W. Devanny, D. Eppstein, and M. Goodrich, "Scheduling Autonomous Vehicle Platoons Through an Unregulated Intersection," sep 2016. 3, 8

[10] P. Fernandes and U. Nunes, "Algorithms for management of a multi-platooning system of ivc-enables autonomous vehicles, with high traffic capacity," in *14th International IEEE Conference on Intelligent Transportation Systems*, pp. 1935–1941, oct 2011. 3

[11] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, "Overview of platooning systems," in *19th ITS World Congress*, jan 2012. 6

[12] T. Kalbitz, C. Becker, and C. Krupitzer, "A Comparison of Approaches for Platooning Management II," tech. rep., University of Mannheim, feb 2017. 6

[13] A. Khodayari, A. Ghaffari, S. Ameli, and J. Flahatgar, "A historical review on lateral and longitudinal control of autonomous vehicle motions," in *2010 International Conference on Mechanical and Electrical Technology*, pp. 421–429, IEEE, sep. 6

[14] J. Zhou and H. Peng, "Range Policy of Adaptive Cruise Control Vehicles for Improved Flow Stability and String Stability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, pp. 229–237, jun 2005. 6

[15] L. Hobert and O. Altintas, "A study on platoon formations and reliable communication in vehicle platoons," tech. rep., University of Twente, Enschede, the Netherlands, jan 2012. 6, 7

[16] D. Swaroop and J. K. Hedrick, "Control of mobile platforms using a virtual vehicle approach," *Journal of Dynamic Systems, Measurement, and Control*, vol. 46, pp. 462––470, sep 1999. 7

[17] A. S. M. Egerstedt, X. Hu, "Constant Spacing Strategies for Platooning in Automated Highway Systems," *IEEE Trans. Autom. Control*, vol. 121, pp. 1777—1782, jul 2001. 7

[18] M. Shida and Y. Nemoto, "Development of a Small-Distance Vehicle Platooning System," in *16th ITS World Congress and Exhibition on Intelligent Transport Systems and Services*, 9 2009. 7

[19] S. Hallé and B. Chaib-draa, "Collaborative driving system using teamwork for platoon formations," in *Applications of Agent Technology in Traffic and Transportation*, pp. 133–151, 2004. 8

[20] R. Hall and C. Chin, "Vehicle sorting for platoon formation: Impacts on highway entry and throughput," tech. rep., Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA, United States, mar 2002. 8

[21] S. Hoshino, "Reactive Clustering Method for Platooning Autonomous Mobile Robots," *IFAC Proceedings Volumes*, vol. 46, pp. 152–157, jun 2013. 8

[22] S. E. Shladover, "Operation of merge junctions in a dynamically entrained automated guideway transit system," *Transportation Research Part A: General*, vol. 14, pp. 85–112, apr 1980. 8, 9

[23] D. Miculescu and S. Karaman, "Polling-systems-based control of highperformance provably-safe autonomous intersections," in *IEEE 53rd Conf. Dec. Control*, pp. 1417–1423, dec 2014. 8

[24] A. Uno, T. Sakaguchi, and S. Tsugawa, "A merging control algorithm based on inter-vehicle communication," in *Proceedings 199 IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems*, pp. 783–787, oct 1999. 9

[25] A. I. M. Medina, N. van de Wouw, and H. Nijmeijer, "Automation of a t-intersection using virtual platoons of cooperative autonomous vehicles," in *IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 1696–1701, sep 2015. 9

[26] K. J. C. Fransen, "A path planning approach for agvs in the dense grid-based *AgvSorter*," Master's thesis, Eindhoven University of Technology, jul 2019. 11

[27] A. M. Law and W. Kelton, *Simulation Modeling and Analysis*. 2000. 44, 45, 46

[28] C. Voorhis and B. Morgan, "Understanding power and rules of thumb for determining sample size," *Tutorials in Quantitative Methods for Psychology*, vol. 3, 09 2007. 45

[29] G. Cumming and S. Finch, "Inference by eye: Confidence intervals and how to read pictures of data," *The American psychologist*, vol. 60, pp. 170–80, 02 2005. 46

List of abbreviations

Abbreviation	Meaning
ACC	Adaptive Cruise Control
AGV	Automated Guided Vehicle
CACC	Cooperative Adaptive Cruise Control
CC	Cruise Control
CTC	Center-To-Center
RoW	Right-of-Way
STS	Surface-To-Surface
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle

Glossary

Term	Definition
Adaptive cruise control	A longitudinal control technology allowing feedback control based on on-board information to maintain a constant safe distant between preceding vehicles.
Automated guided vehicle (AGV)	Autonomous mobile robot.
Available tile	A tile that is available for reservation by an AGV, as opposed to an obstacle.
Benchmark situations	Basic zoomed in scenario's that appear in bigger layouts, such as a crossing, a turn, and a loading point.
Clothoid	A curve whose curvature changes linearly with its curve length, used to create the segments for cutting corners.
Congestion	The state where AGVs have accumulated in a certain area, usually results in delays.
Cooperative adatpive cruise control	A longitudinal control technology allowing both feedback and feedforward control to maintain a constant safe distant between preceding vehicles. Feedback control is based on on-board information, and feedforward is possible due to V2V communication.
Corner entry tile	The tile an AGV enters a turn.
Corner exit tile	The tile an AGV leaves a turn.
Corner turning tile(s)	The tile(s) inbetween the corner entry tile and the corner exit tile.
Cruise control	A longitudinal control technology where a constant user-inputted speed is maintained.
Curved-line platooning	A platooning strategy that allows cutting a corner as a platoon.
Cutting corners	Cutting corners is a turning strategy where a turn is made by following a curve (clothoid segment) without coming to a full stop, as opposed to pivoting.

Term	Definition
Cutting corners decision point	The point before which an AGV needs to decide if it will make the turn by pivoting or by cutting corners.
Cutting corners waiting point	A point defined on a clothoid segment at which an AGV waits if it cannot continue to its next tile.
Deadlock	State where all AGVs in a group are standing still, waiting on another AGV in this group (circular wait).
Dense	Crowded, an AGV system is considered dense when per 100 available tiles 20 AGVs, or more, are in the system.
Destination	A location on the grid where an item can be delivered.
Flexibility	The ability of a system to handle changes in system configuration, such as different layouts, or a different amount of AGVs.
Follower	The AGV that considers a follow-up on an indication given by a leader.
Follow-up	A follow-up can be given by an AGV on an indication, which means the AGV will take over a tile reservation from the leader AGV that gave an indication for this tile.
Grid layout	A possible layout of the AgvSorter system.
Grid-based AGV system	A grid-based AGV system is an AGV system controlled with grid-based control.
Grid-based control	Grid-based control is a control strategy where vehicles can only travel in the directions that are defined by the grid. The grid consists of tiles, with segments allocating allowed driving directions. These tiles are 'virtual', and do not have any physical purpose. A vehicle can only travel over a tile, if it has claimed this tile.
Grid-based platooning	Platooning strategies applicable for grid-based AGV systems as proposed in this system, i.e. straight-line platooning, and curved-line platooning.
Grid-type layouts	A type of layout where the majority of the tiles has two incoming segments and two outgoing segments, as opposed to segment-type layouts.
Heading	The angle between the direction the front side of the AGV is pointed towards, and a world frame.
Indication	If an AGV knows it will leave a tile, because it has already reserved another tile, it can give an indication of leaving to other AGVs.

Term	Definition
Item leadtime	The time it takes an AGV to finish the task of picking up and dropping off an item, after being allocated this task.
Key performance indicator	A measurable value that demonstrates the performance of a system.
Leader	A vehicle that gives an indication.
Lead time percentile	The lead time percentile refers to the 95% percentile of job lead times.
Node	Nodes are points where two or more segments are connected.
Observed platoon length	The number of consecutive AGVs passing a tile as a platoon, as observed by an AGV outside of the platoon also waiting on this tile.
Obstacle	A type of tile that is not available for reservation by an AGV as opposed to available tiles.
Path separation point	The point at which the paths associated with the pivoting strategy and cutting corners strategy start to differ.
Permitted platoon length	The permitted platoon length is the maximum allowed number of consecutive AGVs passing a tile as a platoon.
Pivoting	Pivoting is a turning strategy where a turn is made by getting to a full stop at the center of a tile, and turning on the spot, as opposed to cutting corners.
Platoon control	Platoon control regulates intra platoon interaction.
Platoon coordination	Platoon coordination regulates inter platoon interaction.
Platoon manoeuvres	Platoon manoeuvres are actions that alternate the vehicle composition of a platoon.
Platooning	A platoon is a string of vehicles, accelerating and decelerating together, while maintaining a small inter-vehicular distance.
Platooning action	A platooning action comprises a leader vehicle giving an indication, and a follower vehicle giving a follow-up on this indication.
Right-of-way (RoW) platoon length	The maximum number of consecutive AGVs passing a tile as a platoon, at which an AGV in the platoon still has priority over AGVs outside of the platoon.
Robustness	The ability of a system to handle with local system malfunction.
Scalability	The ability of a system to handle bigger workloads by adding resources (AGVs) to the system.
Segment	A line piece over which AGVs can drive connecting two nodes.
Segment-type layouts	A type of layout where the majority of the tiles has one incoming segments and one outgoing segments, as opposed to grid-type layouts.

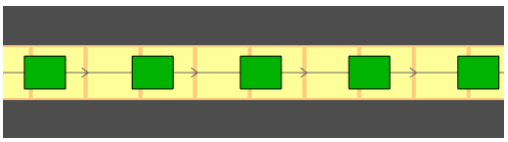
Term	Definition
Segment speed	The maximum allowed speed for an AGV on a given segment.
Source	A location on the grid where an item can be picked up.
Straight-line platooning	A platooning strategy that only allows platooning on straight segments.
String stability	The property string stability ensures that any perturbations of the velocity or position of the leading vehicle will not result in amplified fluctuations to the following vehicle's velocity and position.
Throughput	The amount of tasks finished per time unit.
Tile	An element of the grid.
Traffic control	Traffic control ensures conflict-free driving of AGVs, in following their paths. This includes assuring no collisions or deadlocks occur by controlling AGV speeds.
True platoon length	The number of consecutive AGVs passing a tile as a platoon.
Vehicle-to-infrastructure communication	The wireless exchange of data between vehicles and road infrastructure.
Vehicle-to-vehicle communication	The wireless exchange of data between vehicles.

A: Benchmark scenarios

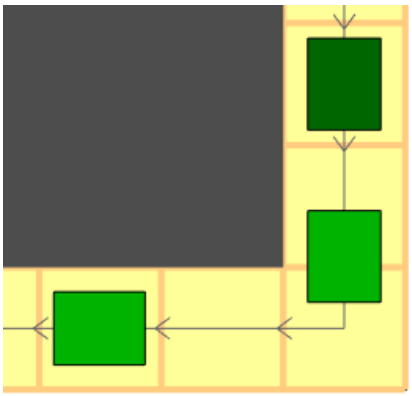
This appendix shows the benchmark scenarios. In Table A.1, the parameter values used in all scenarios are shown. In Figures a-j the scenarios are visualized.

Parameter	**Symbol**	**Value**	**Unit**
AGV length	l_{AGV}	1.50	m
Maximum segment speed	v_{max}	2.50	m/s
Maximum curve speed	v_{curve}	2.00	m/s
Maximum acceleration	a_{max}	2.00	m/s^2
Maximum deceleration	d_{max}	2.00	m/s^2
Rotation speed	ω	0.785	rad/s
Tile size	A_{tile}	2 x 2	m^2
Loading time	t_{load}	2	s

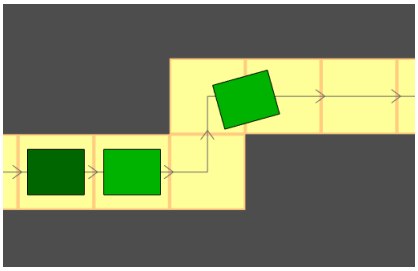
Table A.1: Parameter values for the benchmark scenarios



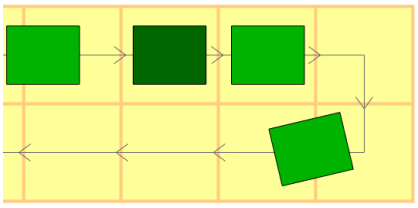
(a): Straight



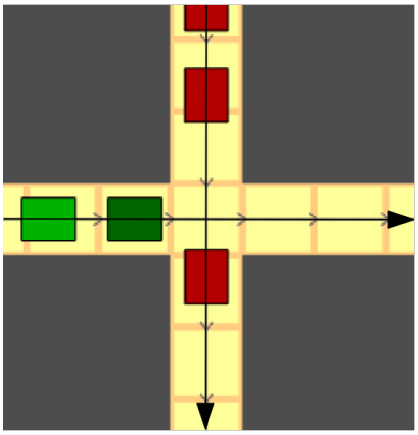
(b): Curve



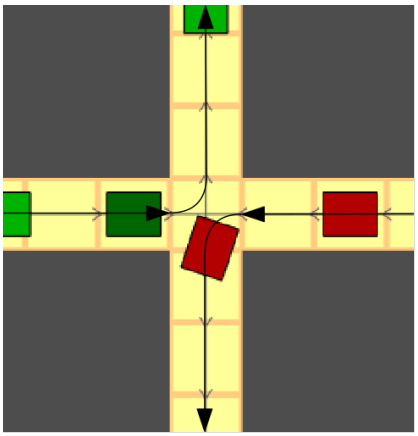
(c): S-curve



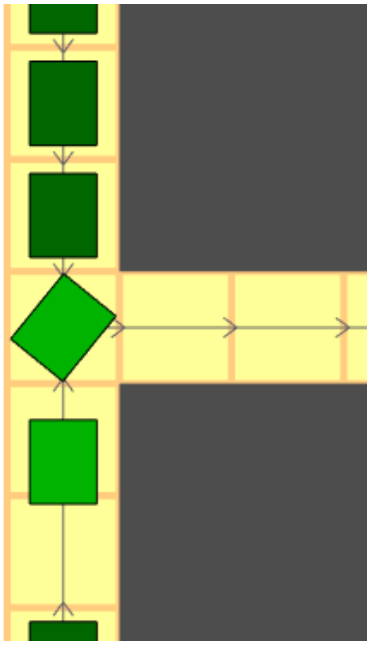
(d): U-curve



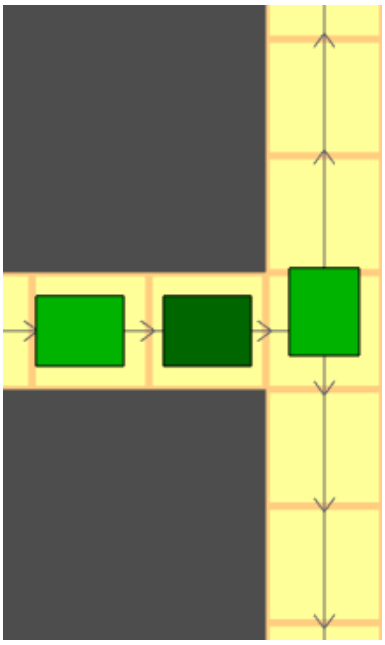
(e): Crossing



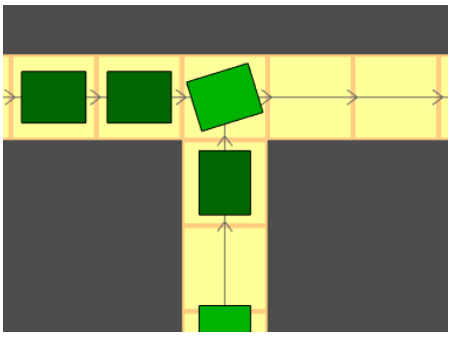
(f): Crossing-turn



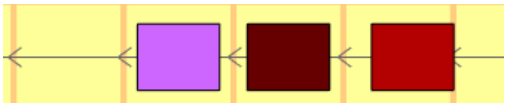
(g): T-merge



(h): T-split



(i): Ramp



(j): Loading

B: System configuration parameters

Tables B.1, B.2, and B.3 show the configuration parameters for the *GridSorter*, *Airport*, and *Tessellation* configuration, respectively.

Parameter	Symbol	Value	Unit
AGV length	l_{AGV}	0.70	m
Maximum speed	$v_{\max}$	1.00	m/s
Maximum curve speed	v_{curve}	0.75	m/s
Maximum acceleration	$a_{\max}$	1.00	m/s^2
Maximum deceleration	$d_{\max}$	1.00	m/s^2
Maximum rotation speed	ω	1.57	rad/s
Loading time	t_{load}	1	s
Unloading time	t_{unload}	1	s

Table B.1: Configuration parameters of the *GridSorter* configuration

Parameter	Symbol	Value	Unit
AGV length	l_{AGV}	1.50	m
Maximum speed	$v_{\max}$	2.50	m/s
Maximum curve speed	v_{curve}	2.00	m/s
Maximum acceleration	$a_{\max}$	2.00	m/s^2
Maximum deceleration	$d_{\max}$	2.00	m/s^2
Maximum rotation speed	ω	1.57	rad/s
Loading time	t_{load}	1	s
Unloading time	t_{unload}	1	s

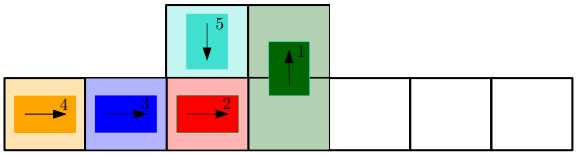
Table B.2: Configuration parameters of the *Airport* configuration

Parameter	Symbol	Value	Unit
AGV length	l_{AGV}	0.70	m
Maximum speed	$v_{\max}$	1.00	m/s
Maximum curve speed	v_{curve}	0.75	m/s
Maximum acceleration	$a_{\max}$	1.00	m/s^2
Maximum deceleration	$d_{\max}$	1.00	m/s^2
Maximum rotation speed	ω	1.57	rad/s
Loading time	t_{load}	1	s
Unloading time	t_{unload}	1	s

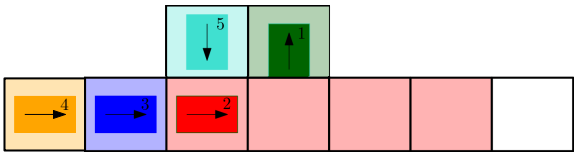
Table B.3: Configuration parameters of the *Tessellation* configuration

C: Straight-line platooning desired behavior

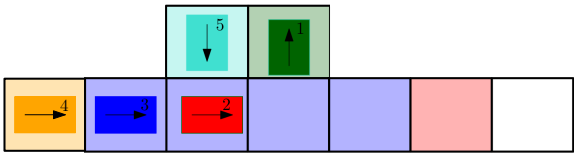
This appendix shows a situation in which the behavior prompted by straight-line platooning is eminent.



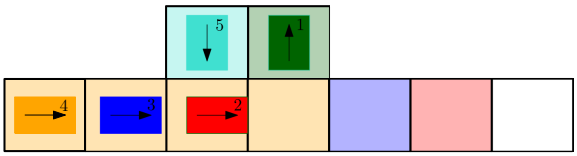
Start situation



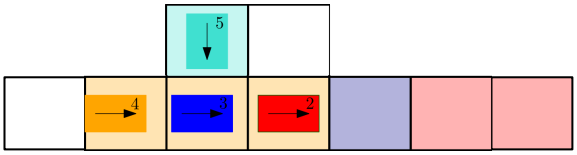
After AGV 1 (green) clears the tile, the AGV 2 (red) can reserve three tiles ahead.



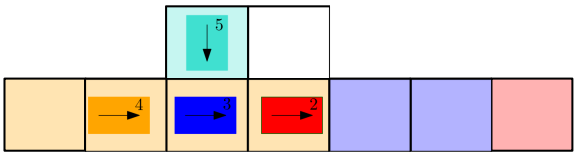
After a small delay, AGV 2 gives indications for its first three reserved tiles, and the AGV 3 (dark blue) follows-up on these indications.



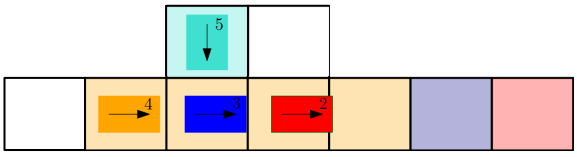
After a small delay, AGV 3 gives indication for its first three reserved tiles, and AGV 4 (yellow) follow-ups on these indications.



AGV 2 reaches the center of its next tile, and can make a new tile reservation.



After a small delay AGV 2 gives an indication for its second-to-last reserved tile, and AGV 3 follows-up on this indication.



After a small delay, AGV 3 gives an indication for its second-to-last reserved tile, and AGV 4 follows-up on this indication. Meanwhile, AGV 5 (cyan) is still waiting.

D: Minimum distance check

This appendix elaborates on the minimum distance check. The output of the minimum distance check is either 'true' or 'false'.

For both the leader and the follower vehicle the time necessary to travel from their current location to their respective critical point is calculated. The follower critical point is defined as the point $0.75 \cdot l_t$ before the center of the turn, and the leader critical point is defined as the point $0.75 \cdot l_t$ after the center of the turn. If the follower takes longer than the leader, the minimum distance check passes. To determine these times, the travel time for all segments between the AGV's current location and the critical point are determined and added up. Determining the time necessary to drive over a segment from one point to the next is an involved step. Given the initial AGV speed, v_i, the current segment's maximum speed, $v_{\max,i}$, the next segment's maximum speed $v_{\max,i+1}$, the maximum acceleration, a, the maximum deceleration d, and the remaining segment length, l there are several velocity profiles, each requiring a different calculation method. We distinguish six profiles, see Figure D.1.

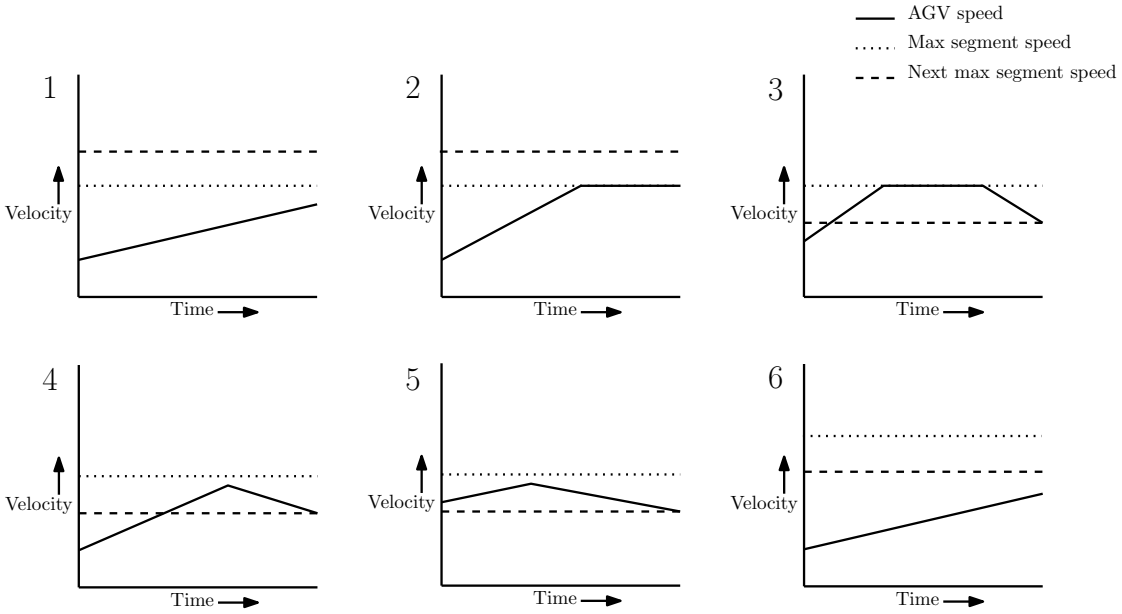


Figure D.1: Plots of the five velocity profiles

1. The next segment's maximum speed is higher than, or equal to current segment's, and before reaching the next segment the current segment's maximum speed is reached. The time necessary to cross such a segment, t_s can be determined by solving Equation D.1 for t_s.

$$l = v_i t_s + \frac{1}{2} a t_s^2 \tag{D.1}$$

2. The next segment's maximum speed is higher than, or equal to current segment's, and before reaching the next segment the current segment's maximum speed is not reached. The time necessary to cross such a segment, t_s can be determined with Equation D.2.

$$\begin{aligned} t_s = t_a + v_{\max,i}(l - (v_i t_a + \frac{1}{2} a t_a^2)), \\ \text{with } t_a = \frac{(v_{\max,i} - v_i)}{a} \end{aligned} \tag{D.2}$$

3. The next segment's maximum speed is lower than the current segment's, and before having to decelerate the current segment's maximum speed is reached. The time necessary to cross such a segment, t_s can be determined with Equation D.3.

$$\begin{aligned} t_s = t_a + t_d + v_{\max,i}(l - (v_i t_a + \frac{1}{2}, a t_a^2) - (v_{\max,i+1} t_d + \frac{1}{2} d t_d^2)), \\ \text{with } t_a = \frac{(v_{\max,i} - v_i)}{a}, \\ \text{and } t_d = \frac{(v_{\max,i} - v_{\max,i+1})}{d} \end{aligned} \tag{D.3}$$

4. The next segment's maximum speed is lower than the current segment's, before having to decelerate the current segment's maximum speed is not reached, and the initial AGV's speed is higher than the next segment's speed. The time necessary to cross such a segment, t_s can be determined by solving Equation D.4 for t_s.

$$\begin{aligned} l = t_m + v_{\max,i+1}(t_s - t_m) + (\frac{1}{2} \frac{a^2}{a+d})(t_s - t_m), \\ \text{with } t_m = \frac{v_{\max,i+1} - v_i}{a} \end{aligned} \tag{D.4}$$

5. The next segment's maximum speed is lower than the current segment's, before having to decelerate the current segment's maximum speed is not reached, and the initial AGV's speed is lower than the next segment's speed. The time necessary to cross such a segment, t_s can be determined by solving Equation D.5 for t_s.

$$\begin{aligned} l = v_{\max,i+1} * t_s + \frac{1}{2} t_m (v_i - v_{\max,i+1}) \\ + (t_s - t_m)(v_i - v_{\max,i+1}) + \frac{1}{2} \frac{a^2}{a+d} (t - t_m), \\ \text{with } t_m = \frac{v_{\max,i+1} - v_i}{a} \end{aligned} \tag{D.5}$$

6. The next segment's maximum speed is lower than the current segment's, and the initial AGV's speed is lower than the next segment's speed, but the next segment's maximum speed is not reached before the end of the current segment. The time necessary to cross such a segment, t_s can be determined by solving Equation D.6 for t_s.

$$l = v_i t_s + \frac{1}{2} a t_s^2 \tag{D.6}$$

E: Calculating bottleneck throughputs

This appendix elaborates on the calculations for the bottleneck throughputs of the benchmark scenarios. In every scenario a certain pattern iterates. The bottleneck throughput can be determined by dividing the number of AGVs passing the scenario in one iteration by the duration of one iteration times the number of incoming lanes. The patterns vary per scenario, but can always be constructed with the following basic pattern elements, or elements that are equivalent.

1. An AGV follows another AGV over a straight lane without platooning
2. An AGV follows another AGV over a straight lane via a platooning action
3. An AGV follows another AGV through a corner without platooning, the turn is made using a pivoting strategy.
4. An AGV follows another AGV through a corner by applying a platooning action.
5. An AGV follows another AGV, and makes a turn by cutting the corner without platooning, the other AGV turns into the opposite direction also by cutting a corner.
6. An AGV follows another AGV over a source/destination tile without platooning, and needs to stop to (un)load.
7. An AGV follows another AGV over a source/destination tile without platooning, and needs to stop to (un)load.
8. An AGV follows another AGV through a corner without platooning, the turn is made using a pivoting strategy. After the turn the AGV needs to come to a standstill.

Calculate inter-vehicle time gaps for basic pattern elements

Next, for all basic pattern elements, the inter-vehicle time gap is calculated. Using this, the time of one scenario iteration can be determined, allowing to determine the bottleneck throughputs.

Element 1: In order to determine the inter-vehicle time gap for two vehicles driving on a straight lane without platooning, first we determine the acceleration time and acceleration distance from standstill:

$$\begin{aligned} t_a &= v_{\max}/2 = 1.25\text{s} \\ s_a &= 0.5 * a_{\max} * t_a^2 = 1.5625\text{m} \end{aligned} \tag{E.1}$$

Now, it is possible to determine the inter-vehicle time gap by dividing the inter-vehicle distance at maximum speed by the maximum speed:

$$\delta t_1 = (v_{\max} * t_a + 1.5 * l_{tile} + 0.5 * l_{\text{AGV}} - s_a)/v_{\max} = 2.125\text{s} \tag{E.2}$$

Element 2: To determine the inter-vehicle time gap for element 2, the inter-vehicle distance at maximum speed is divided by the maximum speed.

$$\delta t_2 = (t_{\text{delay}} * v_{\max} + l_{\text{tile}})/v_{\max} = 1\text{s} \tag{E.3}$$

Element 3: First, the pivoting time and the time necessary to move from standstill to standstill from one tile to a neighbouring tile are calculated:

$$\begin{aligned} t_p &= 0.5\pi/0.7853 = 2\text{s} \\ t_{t2t} &= 0.5 * a_{\max} * t_a^2 = 1.5625\text{m} \end{aligned} \tag{E.4}$$

This allows to determine the inter-vehicle time gap by dividing the inter-vehicle distance after the scenario when both AGVs are at maximum speed by the maximum speed:

$$\delta t_3 = (0.5 * l_{\text{tile}} + 0.5 * l_{\text{AGV}} + v_{\max} * (t_p + t_{t2t} + t_a) - s_a)/v_{\max} = 5.325\text{s} \tag{E.5}$$

Element 4: The distance between two AGVs cutting a corner is defined to minimally be $1.25 * l_{\text{tile}}$ at the critical point. The AGVs drive at maximum speed at the critical point. Therefore the inter-vehicle time gap can be determine using the following equation:

$$\delta t_4 = (1.25 * l_{\text{tile}})/v_{\text{curve}} = 1.25\text{s} \tag{E.6}$$

Element 5: Corner segments are constructed of clothoids. There is no general expression to determine the coordinates along a clothoid from the length across the arc. The *AgvSorter* simulation model uses an approximation for this. A turn consists of three tiles. An AGV can start the turn if the first two tiles are unclaimed. Therefore the inter-vehicle time gap equals the time between the AGV moving from standstill from the first tile to leaving the second tile. This time takes approximately 2.0720 s.

$$\delta t_5 = 2.0720\text{s} \tag{E.7}$$

Element 6: The inter-vehicle time gap of this element 6 can be determined by calculating the time between the first AGV moving from standstill from the source/destination tile, and the second finishing (un)loading. This is done in the following equation:

$$\delta t_6 = (0.5 * l_{\text{tile}} + 0.5 * l_{\text{AGV}} + v_{\max} * (t_{\text{load}} + t_{t2t}) - s_a)/v_{\max} = 5.325\text{s} \tag{E.8}$$

Element 7: The inter-vehicle time gap of element 7 can be determined by calculating the time between the first AGV moving from standstill from the source/destination tile and the second finishing (un)loading. This is done in the following equation:

$$\delta t_6 = 0.2 + t_{t2t} + t_{\text{load}} = 4.2\text{s} \tag{E.9}$$

Element 8: A turn consists of three tiles. The inter-vehicle time gap is determined by calculating the time between moving from standstill from the first tile until leaving the the second time.

$$\delta t_8 = (t_{t2t} + t_p + t_{t2t} - \sqrt{\frac{l_{\text{tile}} - l_{AGV}}{d_{\max}}})/v_{\max} = 5.5\text{s} \tag{E.10}$$

Calculate bottleneck throughputs per scenario

With the inter-vehicle time gaps of the element, it is now possible to determine the bottleneck throughputs for all scenarios for the different strategies. This is done by dividing the number of AGVs in one pattern iteration, over the sum of the inter-vehicle time gaps relevant for the scenario, times the number of incoming lanes. This is done in Tables E.1 and E.2. In case of multiple incoming lanes, the coordination influences the bottleneck throughputs. The *max platoon length* is denoted with n in the tables.

Scenario	**Strategy**	**#AGVs**	**Pattern elements**	**#Incoming lanes**	**Bottleneck throughput [AGVs/hour]**
Straight	Baseline pivoting	1	1	1	$3600/\delta t_1$
	Baseline cutting corners	1	1	1	$3600/\delta t_1$
	Straight-line platooning	1	2	1	$3600/\delta t_2$
	Curved-line platooning	1	2	1	$3600/\delta t_2$
Curve	Baseline pivoting	1	3	1	$3600/\delta t_3$
	Baseline cutting corners	1	3	1	$3600/\delta t_3$
	Straight-line platooning	1	3	1	$3600/\delta t_3$
	Curved-line platooning	1	4	1	$3600/\delta t_4$
S-curve	Baseline pivoting	1	8	1	$3600/\delta t_8$
	Baseline cutting corners	1	8	1	$3600/\delta t_8$
	Straight-line platooning	1	8	1	$3600/\delta t_8$
	Curved-line platooning	1	4	1	$3600/\delta t_4$
U-curve	Baseline pivoting	1	8	1	$3600/\delta t_8$
	Baseline cutting corners	1	8	1	$3600/\delta t_8$
	Straight-line platooning	1	8	1	$3600/\delta t_8$
	Curved-line platooning	1	4	1	$3600/\delta t_4$
Crossing	Baseline pivoting	1	1	2	$3600/2\delta t_1$
	Baseline cutting corners	1	1	2	$3600/2\delta t_1$
	Straight-line platooning	n	1, (n-1)x2	2	$(3600*n)/2(\delta t_1+(n-1)\delta t_2)$
	Curved-line platooning	n	1, (n-1)x2	2	$(3600*n)/2(\delta t_1+(n-1)\delta t_2)$

Table E.1: Determining the bottleneck throughputs for 'Straight', 'Curve', 'S-curve', 'U-curve', and 'Crossing'.

Scenario	**Strategy**	**#AGVs**	**Pattern elements**	**#Incoming lanes**	**Bottleneck throughput [AGVs/hour]**
Crossing-turn	Baseline pivoting	1	3	2	$3600/2\delta t_3$
	Baseline cutting corners	1	4	2	$3600/2\delta t_4$
	Straight-line platooning	1	4	2	$3600/2\delta t_4$
	Curved-line platooning	n	(n-1)x4, 5	2	$(3600*n)/2((n-1)\delta t_4+\delta t_5)$
T-merge	Baseline pivoting	1	3	2	$3600/2\delta t_3$
	Baseline cutting corners	1	3	2	$3600/2\delta t_3$
	Straight-line platooning	1	3	2	$3600/2\delta t_3$
	Curved-line platooning	n	(n-1)x4, 5	2	$(3600*n)/2((n-1)\delta t_4+\delta t_5)$
T-split	Baseline pivoting	1	3	1	$3600/\delta t_3$
	Baseline cutting corners	2	3, 5	1	$(3600*2)/(\delta t_3+\delta t_5)$
	Straight-line platooning	2	3, 5	1	$(3600*2)/(\delta t_3+\delta t_5)$
	Curved-line platooning	1	4	1	$3600/\delta t_4$
Ramp	Baseline pivoting	2	1, 3	2	$3600/(\delta t_1+\delta t_3)$
	Baseline cutting corners	2	1, 3	2	$3600/(\delta t_1+\delta t_3)$
	Straight-line platooning	n	(n-1)x2, 3	2	$(3600*n)/2((n-1)\delta t_2+\delta t_3)$
	Curved-line platooning	2n	1, (n-1)x2, (n-1)x4, 5	2	$(3600*2n)/2(\delta t_1+(n-1)\delta t_2 +(n-1)\delta t_4+\delta t_5)$
Loading	Baseline pivoting	1	6	1	$3600/\delta t_6$
	Baseline cutting corners	1	6	1	$3600/\delta t_6$
	Straight-line platooning	1	7	1	$3600/\delta t_7$
	Curved-line platooning	1	7	1	$3600/\delta t_7$

Table E.2: Determining the bottleneck throughputs for 'Crossing-turn', 'T-merge', 'T-split', 'Ramp', and 'Loading'.

Technische Universiteit
Eindhoven
University of Technology

Declaration concerning the TU/e Code of Scientific Conduct for the Master's thesis

I have read the TU/e Code of Scientific Conduct[i].

I hereby declare that my Master's thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct

Date

20-02-2020

Name

Jeroen Maarten van Willigen

ID-number

0889156

Signature

Submit the signed declaration to the student administration of your department.

[i] See: http://www.tue.nl/en/university/about-the-university/integrity/scientific-integrity/
The Netherlands Code of Conduct for Academic Practice of the VSNU can be found here also.
More information about scientific integrity is published on the websites of TU/e and VSNU