Design of a platooning algorithm for real-time control of a Posisorter's overflow zone

van de Ven, Otto A.M.

Award date:
2022

Report of Graduation Project

Design of a platooning algorithm for real-time control of a Posisorter's overflow zone

MANUFACTURING SYSTEMS ENGINEERING
CONTROL SYSTEMS TECHNOLOGY GROUP

DEPARTMENT OF MECHANICAL ENGINEERING

By	O.A.M. (Otto) van de Ven
	0994268
Supervisor TU/e	dr. ir. J.M. van de Mortel-Fronczak
Supervisor Vanderlande	dr. ir. L. Swartjes

CST 2022.016
Eindhoven, 27th of April 2022

Abstract

Vanderlande is a global market leader for future-proof logistic process automation at airports, in warehouses and the parcel market. This report considers a specific parcel sortation system of Vanderlande, namely the Posisorter. The Posisorter consists of a series of conveyors, where parcels are fed onto the conveyors and later on are pushed off the conveyors at the desired chute. Parcels that are not sorted, if the chute is full for example, are prepared for another round. To prepare parcels for another round, three steps must be performed. First, parcels must be collected and decelerated. Then, a train must be formed with a number of parcels. Last, the train must accelerate before being fed into the system again. Those three steps are performed in a part of the Posisorter called the overflow zone.

In the current design of the overflow zone, some improvements can be made. First, the length of the overflow zone preferably decreases as space is often precious. Second, the train forming currently uses multiple short conveyors that hold a single parcel each. These conveyors need to make a full stop reducing equipment lifetime. To cope with this, a new design of the overflow zone must be made.

A solution is the introduction of platooning in the overflow zone. Platooning comes down to forming a train of vehicles without stopping, and is often used in the context of autonomous vehicles. When parcels are transported by a series of short conveyors, parcels can be considered autonomous vehicles. Platooning combines the three steps that are performed in the overflow zone: deceleration, train forming and acceleration. Therefore, there is a possibility of decreasing the footprint. As parcels keep moving whilst forming a train, the frequent start-stop behaviour is also dealt with.

In the algorithm of (Timmerman and Boon, 2021), trajectories of autonomous vehicles are designed such that a platoon is formed. The design of these trajectories is done by calculating the acceleration and deceleration start times, which computationally is an efficient method. Therefore, the main principle for the design of trajectories can be used for the real-time control of an overflow zone.

In this report, the principle from (Timmerman and Boon, 2021) is used to develop a platooning algorithm for the real-time control of parcels in the overflow zone of a Posisorter. In order to do so, different collections of equations are made to design trajectories based on a parcel's position, velocity and desired departure time. After that, the influence of design parameters on the ability to form platoons of desired length is investigated. From simulation results, it can be concluded that the performance mainly depends on the overflow length and the minimum allowed velocity when forming the platoon. As a footprint ideally is reduced, a trade-off must be made between a smaller footprint and a better performance. The maximum conveyor velocity and acceleration do not have a big influence, investing in powerful and expensive conveyors is therefore not recommended.

Contents

1 Introduction **1**

1.1 Background . . . 1

1.2 Related literature . . . 2

1.3 Contribution . . . 3

1.4 Report organization . . . 3

2 System description **5**

2.1 Type of fixed structure sorters . . . 5

2.2 Posisorter . . . 6

2.3 Overflow zone . . . 9

2.4 Problem of overflow zone . . . 10

2.5 Requirements, constraints and preferences . . . 11

2.6 Key performance indicators . . . 13

3 Platooning for autonomous intersections **15**

3.1 Definition of platooning . . . 15

3.2 Literature research . . . 16

3.3 Platooning as solution . . . 18

3.4 General principle . . . 18

3.5 Additions . . . 19

3.6 Contribution . . . 21

4 Plant model **23**

4.1 Model-based design . . . 23

4.2 Modular design and behaviour of conveyors . . . 24

4.3 Behaviour of parcels on conveyors . . . 24

4.4 Behaviour of collisions between parcels . . . 25

4.5 Poisson arrival rate . . . 27

5 Controller **29**

5.1 Proposed algorithm . . . 29

5.2 Trajectory calculations . . . 35

5.3 Supervisor . . . 45

6 Simulation results **49**

6.1 Validation . . . 49

6.2 Output parameters . . . 50

6.3 Input parameters . . . 50

6.4 Real-time control . . . 55

6.5 Influence of design parameters . . . 56

6.6 Full factorial DOE 59

7 Conclusion and recommendation 63

7.1 Conclusion 63

7.2 Recommendation 64

A Conditions for trajectory design 65

A.1 Supporting equations for ideal trajectory calculation 65

A.2 Supporting equations for decline border calculations 65

A.3 Supporting equations for maximum velocity conditions 66

A.4 Supporting equations for S-curved trajectories calculations 66

B Supporting calculations for collections for trajectory design 69

B.1 Collection 1 69

B.2 Collection 2 69

B.3 Collection 3 70

B.4 Collection 4 70

B.5 Collection 5 71

B.6 Collection 6 71

C Interaction plots 73

Bibliography 75

Introduction 1

In this chapter, an introduction on the project is given. In Section 1.1, the background to the problem as well as an introduction to the problem is given. Then, in Section 1.2, previous research relevant to the problem is mentioned as well as gaps in current literature. After that, in Section 1.3, the contribution of this project to current literature are described. Last, in Section 1.4, the organization of this report is described by briefly discussing the chapters that make up this report.

1.1 Background

Over the last few decades, there is almost no facet of society that has not been affected by the rise of automation; at home, at work or while travelling, people are supported by automated systems. Some examples of these systems in widespread use on a daily bases are ATM machines, the digital navigation system in cars and trucks, and the automatic pilot function in aeroplanes (Parasuraman and Mouloua, 1996).

Vanderlande, a company founded in 1949 in Veghel (NL), contributes to this world of automation by being the global market leader for future-proof logistic process automation at airports. At the moment, the baggage handling systems of Vanderlande are active in more than 600 airports worldwide, including some of the world's largest airports (Vanderlande.com, 2021a). Furthermore, Vanderlande is also a leading supplier of process automation solutions for warehouses and in the parcel market (Vanderlande.com, 2021b). When considering baggage handling and parcel sortation, there are roughly spoken two types of systems that are being developed within Vanderlande.

Firstly, there is the transportation and sortation with conveyor belts or carts attached to a fixed (infra)structure. A few examples of this are the Airtrax, Baxorter, Crossorter, Helixorter, Posisorter and Truxorter. The main advantage of these systems is that both the capacity (over 10,000 items per hour) and reliability (over 99.9%) are relatively high (Vanderlande.com, 2021c). A major disadvantage, however, is the possibility of a long downtime if a (small) part of the system breaks down. Furthermore, the system cannot be easily expanded or downsized when desired.

Secondly, there is the transportation and sortation by automated guided vehicles. The main advantage of this system is the flexibility and scalability, as the number of vehicles can easily be altered. In addition, if a single vehicle breaks down, the rest of the system is (almost) not affected. However, the capacity of the automated guided vehicles is lower compared to the systems with a fixed infrastructure having the same footprint (Vanderlande.com, 2021d).

In this project, the focus is on a fixed sortation system of Vanderlande. More specifically, this project regards the Posisorter shown in Figure 1.1, a sorter used for the sortation of parcels. The system can be decomposed into multiple *zones*, where each zone has its own function. Some examples of zones are the infeed zone, where parcels are fed into the system, the merge zone, where the parcels of multiple infeeds are merged onto a single conveyor and the sortation zone, where the actual sortation takes place. One zone of particular interest is the *overflow zone*, which collects all parcels that leave the sortation zone and have not been sorted. Parcels that enter the overflow zone must be decelerated, accumulated into a train and then released into the merge zone such that they can be transported to the sortation zone for an additional time.

In order to decelerate and accumulate parcels in the overflow zone, currently conveyors with pneumatic lifting components are used. In this way, conveyors can keep a predefined velocity and the pneumatic components lift parcels from the conveyors at the right moment to form a train. Vanderlande wishes to phase out the usage of pneumatic parts in the overflow zone whilst still being able to accept all incoming parcels, decelerate parcels, accumulate parcels into a train and release a train of parcels to the merge zone. A new design of the overflow zone uses a series of short conveyors to accumulate. Each parcel that enters the overflow zone stops on a single short conveyor until a train of desired length is formed. Then, all conveyors accelerate to transport the formed train to the merge zone. Because each parcel comes to a full stop on the conveyors, however, the short conveyors have to start and stop often which reduces conveyor lifetime.

Besides the undesired start-stop behaviour of conveyors, the footprint of the overflow zone is relatively lengthy as multiple long conveyors are used to decelerate the incoming parcels. By using much space, the parcels can be collected in the overflow zone without causing a jam in the sortation zone or colliding with parcels that are already in the overflow zone. As space is often precious, Vanderlande wishes to reduce the footprint of the overflow zone whilst still guaranteeing that all incoming parcels are accepted and collisions will not occur.

1.2 Related literature

In short, Vanderlande wants to design a new overflow zone that is able to fulfil all the above-mentioned functions, has little to no start-stop behaviour to assure a long lifespan of the conveyors and has a small footprint. A possible solution that tackles both problems is the introduction of a control algorithm that combines all separate zones and accumulates parcels whilst moving. This phenomenon of accumulating whilst moving is referred to as *platoon forming* of *platooning* in short. In literature, multiple papers have been published on the platooning of autonomous vehicles and the control of autonomous intersections. How these autonomous vehicles and autonomous intersections relate to the overflow zone is explained in the next paragraph.

First, let's consider autonomous vehicles. When a series of short conveyors are used in the overflow zone, with short meaning not more than twice the length of a parcel, each conveyor only carries one, or possibly two, parcels. Because of this, parcels are more or less able to move independently of each other and can therefore roughly be seen as autonomous vehicles. Second, there is the autonomous intersection: as the parcels that leave the overflow zone are merged in the merge zone with other arriving parcels, time slots at which the overflow zone is allowed to deliver a (train of) parcel(s) at the merge zone are assigned. These time slots can be seen as an analogy of autonomous intersections, with time slots being the same as the green time of a traffic light at an autonomous intersection.

Current literature on platooning of autonomous vehicles and autonomous intersections presents methods to compute trajectories that autonomous vehicles should follow such that a platoon of vehicles arrives at the intersection at the desired time. These methods only compute the trajectory of a vehicle once based on known arrival times of vehicles and assume this trajectory can be followed precisely. When using these methods to actively control autonomous vehicles in a real-world situation, problem arise. For example, trajectories cannot be redesigned in case they are not followed directly or to avoid collisions. In addition, trajectories must possibly be altered to avoid collisions if a train of parcels arrives at the overflow zone. In short, the methods to compute trajectories from literature cannot be used directly to control the overflow zone.

1.3 Contribution

In this project, it is investigated if it is possible to use platooning in the overflow zone with the aim of reducing the start-stop behaviour of conveyors and the footprint of the overflow zone. In order to do so, current knowledge from literature on platoon trajectory design is used to develop a control algorithm that actively and in real-time controls the conveyors. Different collections of equations that can be used to design trajectories are developed, as well as conditions to choose the correct collection of equations based on based on a parcel's position, velocity and desired departure time. This yields the following research question:

> "*Can a platooning algorithm be developed for real-time control of an overflow zone capable of decelerating, accumulating and releasing a train of trays or totes?*"

If platooning can be used, ideally, one wants to find an optimal design of the overflow zone. However, as Vanderlande delivers tailored solutions to customers, there is not a single optimal design. Therefore, a sensitivity analysis can be performed on the design parameters of the overflow zone to find out their effect on the key performance indicators. From this analysis, the results can be used to support decision making for tailored solutions in the future. This yields the second research queation:

> "*What is the influence of design parameters of the overflow zone controlled by a platooning algorithm on the key performance indicators of the Posisorter?*"

1.4 Report organization

In Chapter 2, information on the Posisorter system that is considered in this report is given and the problem with the current setup is described. Then, in Chapter 3, the results of the literature study on platooning are discussed. and the limitations found during the literature study as well as a statement of contributions are described. In Chapter 4 and Chapter 5, the working and validation of the plant and controller model are explained in detail, respectively. In Chapter 6, the results obtained from simulations on the plant model with the designed controller are presented and analyzed. Lastly, Chapter 7 contains a conclusion and states recommendations for future work.

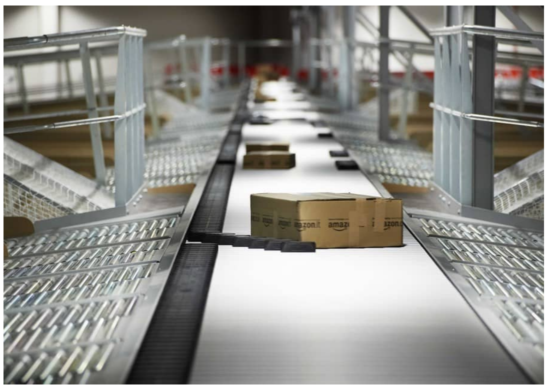


Figure 1.1. Vanderlandes Posisorter system (Vanderlande.com, 2021e).

System description 2

Currently, there is a wide range of sorters used for different applications. As the capacity (the number of goods that can be sorted per time unit) and the handling sizes (the range of goods sizes that can be sorted) vary, different domains may require different types of sorters.

As stated in the introduction, this project focuses on a specific fixed structure sorter system of Vanderlande: the Posisorter. In this chapter, first different types of fixed structure sorters are mentioned and briefly explained to obtain a better insight into the type of fixed structure sorters available and to find out how the Posisorter relates to and differs from other types of fixed structure sorters. Then, the different zones that build up the Posisorter are explained to gain more insights into the working of this sortation system. After that, Section 2.3 goes even more into detail about a zone called the *overflow zone*, as this project mainly regards this zone. Then, in Section 2.4 the problem with the current set-up of the overflow zone is discussed. Last, in Section 2.5 the requirements, preferences and constraints that need to be taken into account when designing a new overflow zone are mentioned and in Section 2.6 the key performance indicators of the Posisorter are described.

2.1 Type of fixed structure sorters

There are different types of fixed structure parcel sortation systems that can be subdivided into two categories: Line sorters and loop sorters (Struik, 2018). Line sorters consist of conveyors that transport parcels over a sortation area. In the sortation area, parcels are diverted by sortation equipment and therefore leave the transporting conveyor. This differs from loop sorters, as parcels on a loop sorter are not transported by conveyors but by carriers on a track. These carriers recirculate and sort the parcel by themselves instead of needing specific sortation equipment in the sortation area. McGuire (2009) mentions the most common ones:

Paddle sorter (line sorter)
Paddle sorters use a pivoting diverter arm, referred to as a *paddle*, to force goods off a conveyor belt or rollers. After sortation, the paddle needs to return to its initial position which takes some time. In the mean time, parcels cannot pass the paddle which results in either a big gap between parcels on high velocity conveyor or a low velocity conveyor. In either way, the capacity is relatively low.

Pusher or puller sorter (line sorter)
A pusher or puller sorter, as the name implies, pushes or pulls the goods off a conveyor at a 90-degree angle. As the goods are not rotated, the width of the pusher or puller must be at least the width of the goods on the conveyor. Therefore, the system is only suitable for relatively small goods. Considering the capacity, the pusher or puller sorter is a mid-range sorter.

Pop-up sorter (line sorter)
A pop-up sorter is placed between conveyors and initially does not divert or even touch the goods. When desired, a belt or rollers pop up to divert the goods. Another type of pop-up sorter is the

steerable roller sorter. Here, the rollers do not actually pop up, but rotate to divert the goods. Pop-up sorters are mid-range sorters regarding their capacity.

Positive sorter (line sorter)
A positive sorter, also called a *shoe sorter*, is different from the sorters mentioned above as the sorter is completely integrated into the conveyor. Instead of a conveyor belt or rollers, sliding shoes are attached to slats that carry the goods. To sort the goods, the shoes traverse diagonally from one to the other side of the conveyor pushing the good of the slats. As the shoes are guided by a mechanical switch, the velocity and therefore the capacity of the positive sorter can be relatively high.

Tilt tray sorter (line or loop sorter)
Tilt-tray sorters can be line or loop sorters. The loop sorter consist of carriers moving on a track, each carrier with a plate on top. The plates on top carry the goods and are able to tilt. When tilting, the goods slide of the carrier making it possible to sort. The line sorters work via the same principle, but instead of carriers moving on a track, conveyors are able to tilt. Tilt-tray sorters generally have a relatively high capacity.

Cross-belt sorter (line or loop sorter)
A cross-belt sorter is similar to a tilt-tray sorter and can also be a line or loop sorter. The loop sorter also consists of carriers moving on a track. Differently from a tilting tray on top, the cross-belt sorter carriers consist of a conveyor belt mounted transverse to the moving direction of the carrier. This conveyor belt carries the goods and by actuating the conveyor the goods on top are sorted. The capacity is, just as for the tilt-tray sorter, relatively high.

2.2 Posisorter

The Posisorter is a positive sorter that is used for the sortation of parcels. As shortly explained above, positive sorters consist of sliding shoes attached to slats that carry the parcel. In this section, the individual zones of the Posisorter are discussed in more detail. In Figure 2.1, an overview of the Posisorter and the different zones can be seen.

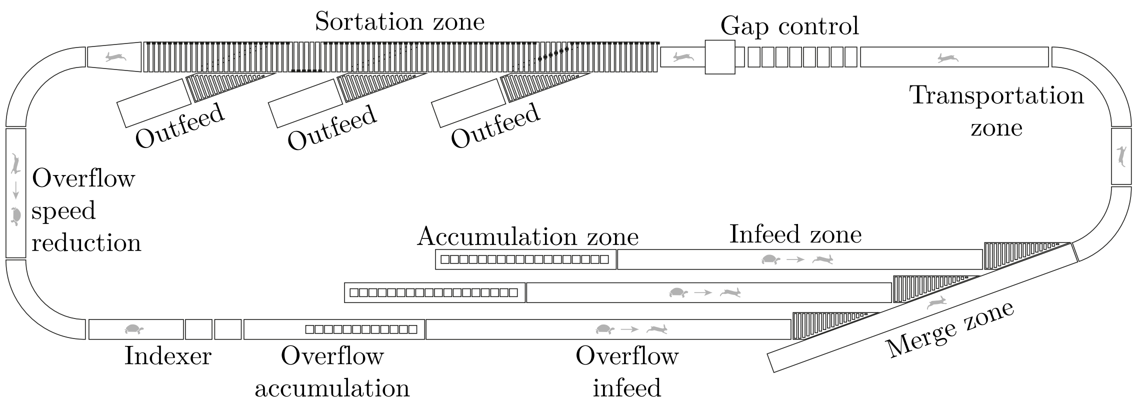


Figure 2.1. Overview of the Posisorter system.

As the moving of parcels on a conveyor belt is often referred to as a *flow* of parcels, the terms *upstream* and *downstream* must also be introduced. Similar to the flow of water in a river, upstream refers to the direction the flow is coming from whereas downstream refers to the direction the flow is moving to.

Before discussing all zones individually, it is also necessary to discuss the load of the sorter. As stated before, the Posisorter is used to sort parcels. In some cases, however, products are not placed directly onto the conveyor belts but placed into a *tray* or *tote*. Trays and totes are fixed-size plastic boxes that contain the product to be sorted. By placing all products into a tray or tote, the system can be designed to handle the fixed-sized trays and totes instead of handling numerous different shapes, sizes and materials. This makes the design of the system more robust, the control of the system more predictable and allows for all shapes and materials to be sorted. In Figure 2.2, the relevant dimensions of the trays and totes are shown. Note that trays can have different heights but the length is always the same.

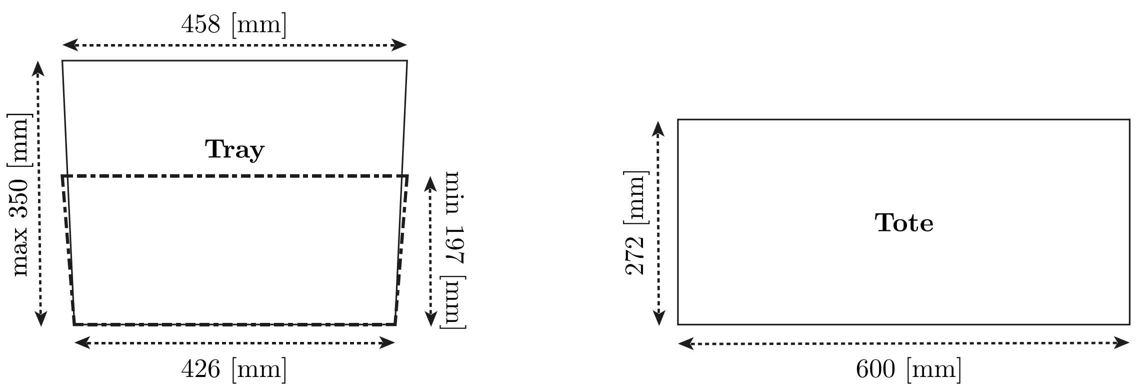


Figure 2.2. Dimensions of the side view of trays and totes.

In case the system holds trays and totes, this is referred to as a *heterogeneous* system. Otherwise, if the system holds either trays or totes, this is referred to as a *homogeneous flow* system. Currently, Vanderlande only uses a homogeneous flow in the overflow zone but heterogeneous flows can be used in the merge and sortation zone. For the remainder of this report, trays and totes will simply be referred to as parcels.

Accumulation zone

The journey of a parcel on the Posisorter starts at the accumulation zone at one of possibly many accumulation fields (but not the overflow accumulation zone). Here, parcels are loaded onto a conveyor and accumulated into a train. The main reason for accumulation is to increase the system throughput, which is the number of parcels sorted per time unit. Why a high system throughput is desired is explained later, but how accumulation relates to throughput can be explained by

$$\mu = \rho * v \tag{2.1}$$

where μ is the throughput [parcels/time unit], ρ is the parcel density on a conveyor [parcels/unit of length] and v is the conveyor velocity [unit of length/time unit]. In Equation 2.1, it can easily be seen that if one wants to increase the throughput, this can either be done by increasing the parcel density on conveyors or by increasing the conveyor velocity. As the latter is not always possible, the parcel density in the merge zone is increased by forming a train in the accumulation zone first.

One could argue that the density in the merge zone can also be increased by releasing single parcels to the merge conveyor in such a way that the merge conveyor is completely filled. There is however a safety margin required between parcels from different infeeds to assure collisions will not occur. These safety margin gaps reduce the parcel density on the merge conveyor significantly when releasing single parcels. When releasing trains of parcels, however, the safety margin only has to be considered between trains. Therefore, the density of the merge zone can be increased and with that the system throughput. Obviously, larger trains need more costly space for accumulation, therefore a trade-off must be made.

Infeed zone
The infeed zone is responsible for transporting, accelerating, and releasing a finished train of parcels onto a single high-velocity conveyor in the merge zone.

Merge zone
The merge zone is responsible for scheduling which infeed may release a train at what time. As the number of infeeds increases, the algorithm used to control the merge zone can become more complex. The merge zone works by dividing the conveyor into virtual windows large enough to store a train. The infeeds can reserve these windows when they have a complete train, and reserving works by the first-come-first-serve principle.

Transport zone
When the parcels are merged, they need to be transported to the sortation zone. Therefore, transporting conveyors are used. It is important that parcels are kept aligned and in a train.

Gap control
Before all parcels can be sorted, a check takes place whether there is enough space between consecutive parcels. As in the next zone, the sortation shoes will slide from one to the other side, it is important that the last shoe assigned to a parcel does not interfere with the front of the consecutive parcel. Therefore, if there is not enough space, the gap control section makes sure a gap is created.

Sortation zone
Next, the actual sortation takes place. Here, shoes that move diagonally are used to push a parcel off the sortation conveyor into the outfeed zone. In Figure 1.1, it can be seen that shoes (black blocks) push a parcel to an outfeed.

Outfeed zone
The journey of a parcel ends at one of possibly many outfeeds. Here, the sorted parcel is queued and waits to be picked up.

Overflow zone
Lastly, there is the overflow zone. All parcels that have not been sorted are collected in the overflow zone, slowed down, accumulated and eventually merged in the merge zone again. Possible reasons for parcels not to be sorted are that the outfeed is full or that the barcode on a parcel in parcel could not have been read. As this project mostly regards the overflow zone, it is elaborated on in more detail in the next section.

2.3 Overflow zone

The overflow zone is responsible for collecting all unsorted parcels, reducing the velocity, accumulating them and releasing them onto the high-speed merge conveyor again. In order to do so, the overflow zone is subdivided into four components: *speed reduction*, *indexing*, *accumulation* and *infeed*. These parts are discussed in more detail below.

Speed reduction

The first step in the overflow process is the reduction of velocity. Parcels that leave the sortation zone have a high velocity, but a low velocity is needed for accumulation. This can also be explained by Equation 2.1: to form a train and with that to increase the density of parcels, the velocity must be reduced at a constant throughput. The exact values of the high and the low velocity depend on the control mode, but the high velocity lies somewhere around 3.5 [m/s] whereas the low velocity lies somewhere around 1.5 [m/s]. The speed reduction is performed by a series of conveyor belts decreasing in speed. Each conveyor belt is relatively long (6-8 meters) and the speed reduction between two consecutive conveyor belts is relatively low (approximately 0.5 [m/s]).

To reduce the speed from 3.5 [m/s] to 1.5 [m/s], for example, approximately 24-32 meters of conveyor belts would be needed. The reason for this relatively lengthy speed reduction process is to allow for a *burst* of parcels to enter the overflow zone. A burst is a short period in which the throughput of parcels is relatively high if, for example, an entire train enters the overflow zone. In such an instance, the first conveyor of the speed reduction component must be able to hold an entire train before slowing down, as slowing down earlier would prevent the last parcel(s) of the train to leave the sortation zone. The latter would cause the entire system to slow down or even to stop completely, which reduces the system's capacity.

The speed reduction component does not use a control algorithm to allow for a smooth, slip-free transition between conveyor belts. The parcels simply leave the upstream conveyor belt and, with some slip and by pushing other parcels, enter the consecutive downstream conveyor belt.

Indexing

The next step is the indexing, comparable to the gap control as discussed before. As the velocity decreases, the density on the conveyor increases. To make sure there is enough space in between parcels to accumulate, a gap must be created between parcels that have no or a too small gap in between. The distance between parcels is now determined by a proximity sensor in the indexer that detects if a parcel is present. If a parcel leaves the indexer and the consecutive parcel is detected by the proximity sensor too soon, the conveyor belt of the indexer slows down or even stops for a small time period to ensure the distance between parcels is large enough. This results in many starts and stops of the indexing conveyor belt.

Accumulation

When the parcels are slowed down and the required gap between them is present, the accumulation takes place. In the accumulation zone, trains of parcels are formed to increase the density as explained in Section 2.2. This is currently done in one of two ways:

> The first way of accumulating is by using a so-called *multibelt* and pneumatic parcel lifters. Parcels are transported over the multibelt, a conveyor system consisting of multiple long but small conveyor belts in parallel. In between those conveyor belts, pneumatic parcel lifters are positioned. As a parcel approaches the right position in the train, it is lifted from the

conveyor belt. In this manner, the electronically actuated conveyor belt does not need to change velocity and the parcels stop moving. When a train is formed, the pneumatic lifters are lowered simultaneously making sure all parcels start to move to the infeed at the same time.

Another way of accumulating is by so-called *shortbelts*. Shortbelts are short conveyor belts of approximately the length of a parcel. This means that each short belt is able to hold a single parcel. A number of shortbelts are positioned in a row and used to form a part of the train. The incomplete train then leaves the shortbelts and is moved to a conveyor belt that is able to hold the incomplete train. This process is repeated multiple times until a train of desired length is formed. This train is then moved to the infeed.

Infeed

The last part is the infeed. As discussed in Section 2.2, the infeed communicates with the merge zone in order to release the train of eighteen parcels onto the high-velocity conveyor. An addition to the normal infeeds is that the overflow infeed has priority and can therefore always claim a window on the merge conveyor. This is essential, as the overflow zone must accept all incoming parcels without delay. If the overflow zone cannot accept any more parcels from the sortation zone, the entire system needs to stop. Since all parcels that enter the overflow zone must also leave it and the consequences of a full overflow zone are severe, the overflow infeed has priority when it comes to claiming windows on the merging conveyor.

Control modes

The four components operate as described above when the overflow zone is in *normal* mode. There is however another mode: the *emergency overflow* mode. In this mode, the parcels are not accumulated. The overflow zone simply transports the parcels from the sortation zone back to the merge zone at low velocity. To avoid any collisions, it is not possible for other infeeds to release any parcel onto the merge conveyor. The emergency overflow mode has two goals:

- First, the emergency overflow mode is used to prevent the complete system from stopping when the overflow zone impends to overflow itself.
- Second, the emergency overflow mode allows the system to start again after a full system stop without manual intervention.

The emergency overflow mode can be triggered in two ways. Firstly, when the throughput at the entrance of the overflow zone exceeds a certain threshold for a specified period of time. When this happens, it cannot be guaranteed that all parcels that approach the overflow zone can leave the sortation zone. Secondly, when the overflow zone is filled for 75% by parcels. Again, when this happens it cannot be guaranteed that all parcels that approach the overflow zone can leave the sortation zone. Ideally, the emergency overflow mode is not activated often.

2.4 Problem of overflow zone

Now the general working of the Posisorter and the overflow zone are explained, the problem with the current design of the overflow zone can be discussed. First off, Vanderlande wishes to discontinue the use of pneumatic systems as these systems are generally noisy, have a high energy loss and are not sustainable. Unfortunately, the most common way of accumulating in the overflow zone is by using a multibelt with pneumatic lifters as mentioned in Section 2.3. The second problem with the current design regards the indexer, that shows a lot of start-stop behaviour which reduces the lifetime of the indexer's conveyors. The alternative of the multibelt design with an indexer, the shortbelt design as also mentioned in Section 2.3, would fulfil the wish of not using pneumatic systems. However, the shortbelt design comes with some problems of its own.

The shortbelts as they are used currently in the overflow zone also show a lot of start-stop behaviour, where the shortbelts start to accelerate from standstill and decelerate until standstill frequently. This behaviour reduces the lifetime of the shortbelts, and therefore also not ideal. Furthermore, shortbelts use a powerful motor which is expensive, noisy and uses a lot of energy.

Vanderlande has come up with over a dozen physical design concepts for the overflow zone, however, they all either experience the same start-stop behaviour as the shortbelts do now or make use of pneumatic subsystems. It must be noted that a few concepts actually did not experience undesired start-stop behaviour and also did not use pneumatic subsystems, however, these concepts do not meet the requirements of the overflow zone as mentioned in Section 2.5 and are therefore not considered. An example of this is a design that stacks arriving parcels on top of each other in a rack. Although this does not use pneumatic parts or experience a lot of start-stop behaviour, such a design cannot be used to release a train of parcels at the same time.

Another wish of Vanderlande is to decrease the footprint of the overflow zone, as the footprint is relatively large at the moment, occupying precious space. This is mainly due to the fact that the four sub-components of the overflow zone (speed reduction, indexing, accumulation and infeed) are not communicating with each other and trajectories of parcels are not planned. Because of that, the speed reduction component is made long enough to receive an entire train, the accumulation conveyor is able to form and hold an entire train and the infeed is able to hold and send out an entire train which results in a large footprint.

Now, Vanderlande is looking for a new design of the overflow zone to ensure a solid lifetime, no use of pneumatic systems, a smaller footprint and taking all other requirements, constraints and preferences into account.

2.5 Requirements, constraints and preferences

Before designing a new overflow zone, it is important to define the requirements, constraints and preferences of the zone. Requirements describe *must-haves* that stakeholders define, constraints describe real-world boundaries or limits one has to comply with, and preferences describe *nice-to-haves* that stakeholders define (Liaskos et al., 2010). As preferences do not need to be met, they can help in producing a ranking among acceptable plans or designs (Samarati et al., 2018) and therefore they do not have to be measurable. In this section, the requirements, constraints and preferences of the overflow zone are clarified.

Requirements

1. The overflow zone must have a capacity of at least 10% of the system's capacity.
2. The overflow zone must be able to decelerate and accumulate the parcels in trains of eighteen parcels, as the default window size on the merge conveyor is designed for trains of eighteen parcels.
3. The overflow zone must be able to release a train of eighteen parcels simultaneously, as the default window size on the merge conveyor is designed for trains of eighteen parcels.

4. The overflow zone must be able to release an unfinished train with less than eighteen parcels simultaneously, as there might arise situations in which it is not possible to form a complete train.
5. The overflow zone must be able to enter the emergency overflow mode at any time, to prevent the entire system to stop.
6. The overflow zone must not contain any pneumatic parts, as the use of pneumatic parts is expensive and not sustainable.
7. The overflow zone must never overflow, as this may force the entire system to stop which reduces the system's capacity.
8. The overflow zone must be able to handle a burst of eighteen parcels with no gap in between.
9. The overflow zone must be robust: no possible inflow of parcels into the overflow zone may cause the system to get stuck or cause collisions between parcels.

Constraints

1. A tray has a bottom length of 426 [mm], a top length of 458 [mm] and a height between 197 and 350 [mm].
2. A tote has a bottom and top length of 600 [mm] and a height of 272 [mm].
3. Conveyors have a predefined bound on velocity and acceleration based on the conveyor used, with the exact value depending on the conveyor used.

Preferences

1. The overall system throughput must be as high as possible, meaning that the density on the merge conveyor must be as high as possible while the overflow zone must be able to release parcels on the merge conveyor when desired.
2. The parcels must be handled with care, both for the lifetime of a tray/tote as well as for the content inside of the tray/tote.
3. The overflow zone must be sustainable such that, with minimum changes, the system can also be used for parcels of different shapes and forms.
4. The footprint of the overflow zone must be as small as possible.
5. The total costs of the overflow zone must be reduced as much as possible.
6. The control algorithm of the overflow zone can be executed fast enough to work in real-time.
7. The start-stop behaviour of conveyor belts in the overflow zone must be minimized.
8. When stopping is required, the stopping positions must be spread as much as possible such that all conveyors have the same number of stops.

2.6 Key performance indicators

To verify the performance of the design that follows from this project, a measurable value must be given to that design. This can be done with key performance indicators. In this section, these key performance indicators are discussed.

System throughput

Ultimately, the system throughput is the most important key performance indicator the overflow zone has impact on. The system throughput determines how many parcels can be sorted in a period of time. It is one of the most important system parameters for customers of Vanderlande, as customers generally want to sort as many parcels as possible in as little time as possible. In addition, Vanderlande assures a certain capacity for their sortation systems, the design of a new overflow zone should not cause the capacity of the entire system to decrease. The throughput of the overflow zone itself, however, is not that important as a higher throughput of the overflow zone does not necessarily lead to more parcels being sorted per time unit.

In order to maximize the system throughput, the parcel velocity and density in the sortation zone must be as high as possible. The velocity is bounded by hardware constraints, but the density is determined in the merge zone. As the merge zone works by reserving windows of a fixed length, the overflow zone aims at realising trains of parcels to the merge zone equal to the length of the window. The closer a train leaving the overflow zone is to the fixed window length, the higher the density on the merge zone and therefore the higher the system throughput.

The difference in length between a train of parcels released to the merge zone and the fixed window length of the merge zone can therefore be used to define how the overflow zone affects the system throughput.

Footprint

Secondly, the footprint is an important performance indicator. Often, the space needed for a sorter system is costly for the customers of Vanderlande and therefore the footprint needs to be minimized. However, the decrease in footprint should not affect the system throughput much in a negative way.

To illustrate, one can shorten the overflow zone such that there is no space to accumulate parcels. In that case, the footprint decreases significantly. The density of parcels in the merge zone, however, decreases. Furthermore, single parcels are released to the merge zone instead of trains. Hence, the system throughput will ultimately decrease if there is no room for accumulation in the overflow zone.

The footprint of the overflow zone follows directly from the layout that is used in the model, however, it is more important to consider the footprint based on the system throughput. For example, if decreasing the footprint by a factor two yields a system throughput decrease of factor four, this would probably not be recommended. On the contrary, if decreasing the footprint by a factor of two decreases the system throughput one-and-a-half times, this might be recommended.

The number of conveyors used in combination with the length of conveyors can be used to define the footprint of the overflow zone.

Power consumption
Lastly, the power consumption of the equipment used for the design of the overflow zone is considered. The power consumption of the equipment is mainly based on the motor that is used to drive the conveyors. A more powerful motor is generally more expensive, has a higher energy usage and makes more noise compared to a less powerful motor. Note that ultimately the total costs need to be as small as possible while the system throughput must be as high as possible, however, it would be impossible to define the total costs in a general case. The total costs depend, for example, on the price per square meter of the building area and on the energy price which may differ a lot when considering different cases. In addition, the profit obtained when having a higher system throughput depends on a large range of factors and may differ a lot between different companies.

Since defining the total costs is not possible for a general case, it is chosen to determine the energy usage of the equipment used based on the system throughput. As the energy usage directly tells something about the equipment costs and noise level, the power consumption is a good performance indicator.

The maximum velocity and acceleration have a big influence in the power consumption of a conveyor, therefore these parameters can be used to define the relative power consumption of the overflow zone.

3 Platooning for autonomous intersections

As stated in Section 2.4, Vanderlande is looking for a new design of the overflow zone with little to no stopping and a small footprint whilst still accumulating parcels into a train. The accumulation of parcels into a train without stopping has many overlaps with the phenomenon of platooning for autonomous intersections, therefore a literature study has been performed on this topic.

In this chapter, first, the definition of platooning is explained. After that, the connection between autonomous intersections and the overflow zone is clarified. Then, a literature study is presented that discusses relevant papers regarding the problem as defined in Chapter 2. Thereafter, the limitations found in the literature study and the additions needed to cope with them are shown, followed by the contribution to literature.

3.1 Definition of platooning

Before jumping into the advantages and algorithms of platooning for autonomous intersections, it must be defined what platooning actually is. Bergenhem et al. (2012) define platooning as:

> "*A collection of vehicles that travel together, actively coordinated in formation.*"

Kalbitz (2017) adds three aspects to this definition. First, the distance between consecutive vehicles is relatively small while safety restrictions must not be violated. Second, the vehicles in a platoon behave similarly and therefore form a string of vehicles. Third, not all vehicles require the same level of automation. Kalbitz (2017) defines platooning as follows:

> "*Platooning is a string of fully or partly automated vehicles, which drive in a close spacing behind each other without violating safety restrictions.*"

In literature, platooning is often discussed in the context of vehicles like cars or trucks on a long road. Bergenhem et al. (2012), for example, mention five current projects that deal with vehicle platooning on a highway. But how does this relate to the overflow zone? To start with, vehicles can be presented by parcels on a conveyor. However, there is one major difference between the overflow zone and ordinary platoon forming; the overflow zone has a limited length before the parcels must enter the merge conveyor. Because of that, ordinary platoon forming algorithms that are designed to work when there is plenty of road ahead to form a platoon are not very applicable in this situation.

Fortunately, numerous papers have been written on autonomous traffic intersections of which multiple in combination with vehicle platoons. An autonomous intersection is an intersection able to communicate with autonomous vehicles, in order to increase the throughput of the intersection and reduce the mean delay experienced by vehicles.

In papers on autonomous intersections, *polling systems* are used to determine when a certain platoon may cross the intersection. A polling system can be defined as a system that consists of a number of queues, attended by a single server in a fixed order (Boon et al., 2011). Since the merge zone can also be considered a polling system, vehicle platoons crossing an intersection can be considered an analogy of the overflow zone.

Moreover, one of the main goals of autonomous intersections is to increase the throughput of the intersection. Higher throughput of the intersection directly means that the utilization of the intersection, the time the intersection is used, is increased. As the intersection can be represented by the merge zone and the main objective of the merge zone is to have high density, the papers on autonomous intersections can definitely be helpful for this project.

3.2 Literature research

One of the first papers written about autonomous intersection management is Dresner and Stone (2008). In their paper, it is stated that lots of research has been done on self-driving vehicles on open roads, as there is little to no velocity difference between vehicles. It is also stated that at the time of writing, few, if any, research has been done on autonomous intersection management. The paper proposes a first-come-first-serve methodology for tile reservation, where an intersection is divided into multiple tiles. Autonomous vehicles that approach the intersection do not change velocity anymore and based on their current velocity send a reservation request for the desired tiles at the desired time. A global intersection manager can either grant or deny the request, based on tiles being reserved or free. If the request is denied, the vehicle decelerates and requests the same tiles again for a different time. Although Dresner and Stone (2008) do not consider platooning in their paper, it is used as a basis for most papers that are discussed later in this report that do include platooning. In addition, an algorithm that allows for communication between vehicles and tile reservation is presented. This algorithm can possibly be used for the control algorithm of conveyors based on reservations in the merge zone.

Lee and Park (2012) extend the work of Dresner and Stone (2008) by introducing a non-linear constrained optimization problem for a better trajectory of automated vehicles crossing an intersection without colliding. To do so, Lee and Park (2012) define a possible collision window. The possible collision window is a time window in which more than one vehicle is present at the intersection. By minimizing the time a vehicle is in the possible collision window and taking constraints like acceleration and velocity bounds into account, a safe trajectory can be defined. Remarkable is that the paper also introduces a positive minimal velocity, as it is not desired for vehicles to stop completely. In this way, the total stopped delay times have been reduced by 99%. As the start/stop behaviour of conveyors is undesired in this project, the work of Lee and Park (2012) is be useful.

One of the first, or maybe the first, to introduce platooning to autonomous intersections is Tachet et al. (2016a). In their paper, they also extend the work of Dresner and Stone (2008) by first introducing a theoretical framework to assess the performance of autonomous intersections. Then, an algorithm to form a platoon (actually called a *batch* in the paper) and an algorithm to control the platoon to safely cross the intersections are proposed. In Tachet et al. (2016b) the algorithm is discussed in more detail. From simulations, it is concluded that the average delay of vehicles is significantly lower compared to scenarios in which no platoons are formed. Therefore, the capacity of an intersection increases significantly when forming a platoon. As platoon forming based on a polling system is considered in this paper which represents the overflow zone really well, the algorithm used in Tachet et al. (2016a) might be useful.

The papers on autonomous vehicle intersections that resemble the overflow zone really well are the ones of Miculescu and Karaman (2014, 2015, 2020). In their papers, Miculsecu and Karaman describe autonomous ground vehicles that move in a single lane towards a two-way crossing. As a single lane is considered, the only manoeuvre that vehicles can perform is acceleration and deceleration, so no sideways movements. A polling system is used to assign time windows to the lanes in which the vehicles on one of the two lanes may cross the intersection. In order to achieve high throughput with little delay, *smart platooning* is introduced. Smart platooning is defined as the effective clustering of vehicles. This is done by the global `MotionSynthesize` algorithm, which generates a trajectory for each vehicle minimizing the distance to the intersection whilst maintaining a safe distance to the vehicle in front and making sure the intersection is crossed at maximum velocity. From simulation results, it can be concluded that the proposed algorithm would decrease the delay at an intersection significantly compared to an ordinary traffic light setup. Again, as platoon forming based on a polling system is discussed in this paper, the algorithms used might be useful.

Timmerman and Boon (2021) extend the work of Miculescu and Karaman (2014, 2015, 2020) and Tachet et al. (2016a, 2016b) by improving the platoon forming algorithm. Where the other papers make use of optimization solvers to determine the vehicle's trajectory, Timmerman and Boon (2021) proposes an alternative to the `MotionSynthesize` algorithm without needing a solver. By using only closed form expressions that are mentioned in the paper, this algorithm is faster and therefore more applicable for real-time usage. Once again, as platoon forming based on a polling system is discussed in this paper, the algorithms used might be useful.

Paper path planning

Besides platooning and autonomous intersections, paper path planning in printers as described in (Swartjes, 2012) can also be considered an analogy of the overflow zone. Here, parcels can be represented by paper sheets, the conveyors can be represented by pinches that move the paper and the availability on the merge zone can be represented by the availability of different modules within the printer. In his research, Swartjes (2012) converted a paper path planning problem into a one-dimensional problem with an objective function, equality constraints and inequality constraints. Then, using a non-linear solver, the optimal velocity profile for minimal processing time, minimal energy usage and a combination of both is determined. Assumptions have been made on homogeneousness (there is only one type of sheet) and cyclicity (the dynamics of every sheet is the same) and therefore interaction between sheets is neglected. As these assumptions cannot be made for the overflow zone since the inflow rate is variable, the available time windows in the merge zone are variable and the load may consist of parcels, the proposed optimization problem does not comply with the problem at hand. The ideology of converting the problem into a one-dimensional problem with an objective function and constraints might be of interest.

(Alirezaei et al., 2012) came up with a method of using Max-plus algebra for the optimal scheduling of multiple sheets in a printer. This yields a higher utilization of the printer components and increases the number of papers that can be printed per period of time. When considering the scope of the project, the objective is not to come up with a new scheduling algorithm for the merge zone by maximizing the utilization of the merge conveyor. Moreover, the paper does not mention *how* papers must move but only *when* a sheet of paper must be at which component. As there is no information on the trajectory itself, the paper of Alirezaei et al. (2012) will not be of use directly but the scheduling algorithm is useful to obtain insights in how scheduling in the merge zone can be improved.

3.3 Platooning as solution

A promising idea to cope with the current problems is to merge all four sub-components (speed reduction, indexing, accumulation and infeed) into a single component with a number of short conveyors that executes the function of the sub-components. In order to comply with the requirements, constraints and preferences as mentioned in Section 2.5, a control algorithm that controls conveyors is needed that makes sure all approaching parcels are able to enter the overflow zone, slowed down, accumulated and transported in a train to the infeed with minimal stopping. In addition, when stopping is required, the stopping positions must be spread such that all conveyors have approximately the same number of stops instead of having one or a few conveyors that always stop in case a stop is required.

By actively controlling the four sub-components, communicating between them and designing trajectories, the sub-components can be replaced by a single component that executes the function of all sub-components. Now the speed reduction, accumulation and infeed component all must be able to accept an full train. By merging the sub-components, the overflow zone does not have to be able to hold three full trains of parcels allowing for a decrease in footprint.

3.4 General principle

For the generation of trajectories in the overflow zone, the method used by Timmerman and Boon (2021) is used as a basis. Here, trajectories of autonomous vehicles are generated by determining times in the near future at which a vehicle must start and stop accelerating and decelerating. This method is explained based on Figure 3.1.

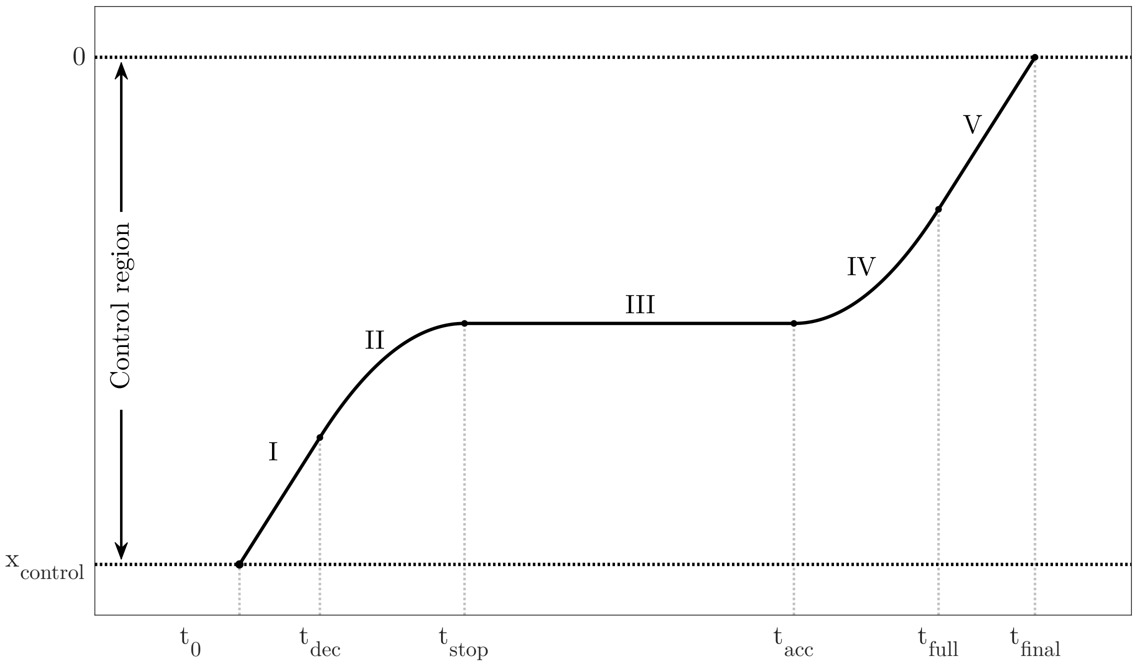


Figure 3.1. Main idea of closed-form trajectory generation by Timmerman and Boon (2021).

In Figure 3.1, the trajectory of a single parcel from position x_{control} to 0 can be seen starting at t_0. As the trajectory is determined between x_{control} to 0, this is where the velocity of the vehicle is controlled and therefore referred to as the *control region*. The trajectory can be cut into five parts (I-V). In the first part, part I, the vehicle holds its initial maximum velocity until t_{dec}. Then, between t_{dec} and t_{stop} the vehicle decelerates with maximum deceleration until the vehicle has come to a full stop. The vehicle then waits before accelerating again based on the chosen final time t_{final}. At t_{acc}, the vehicle accelerates with maximum acceleration until the vehicle reaches its maximum velocity at t_{full}. Finally, during part V, the vehicle moves with maximum velocity to complete the trajectory.

The determination of t_{dec}, t_{stop}, t_{acc} and t_{full} is done by closed-form expressions, based only on vehicle properties like maximum velocity and acceleration, some initial values like the initial velocity and distance to cover and several desired output values like the final velocity and desired end time. Because of that, there is no need for solvers and the trajectory can be determined fairly quickly making it usable for real-time usage (Timmerman and Boon, 2021).

3.5 Additions

Unfortunately, the algorithm described in Timmerman and Boon (2021) only works as desired in a general case when all parameters are bounded relatively strictly. In addition, multiple assumptions have been made that do not apply in the overflow zone. Together, the boundaries and assumptions that have been made form the following limitations that need to be dealt with. The following additions are done to deal with those limitations.

Trajectories can be designed without knowing future arrival times.
In order to determine the trajectories of multiple vehicles that are formed into a platoon, currently the arrival times of all vehicles must be known beforehand. Based on a list of arrival times at x_{control}, the trajectories of all vehicles are designed at once. For the overflow zone, this leads to undesired layout designs of the system as is explained in the next paragraph.

A parcel to enter the overflow zone can first be noticed after it has left the sortation zone. The reason for this is that, if a parcel has not left the sortation zone, it will not be certain if it will still be sorted at the end of the sortation zone or not. After a parcel has left the sortation zone, it is transported with high velocity to the overflow zone where it is slowed down. If a parcel is not allowed to slow down in between, and all parcels that form a platoon must have left the sortation zone before the first parcel may enter the overflow zone, a lengthy transportation zone is needed between the sortation zone and the overflow zone. Only if this transportation is long enough, the arrival time at the overflow zone can be determined for all parcels to form a platoon. A long transportation zone in turn reduces the key performance indicator of a small footprint. Therefore, trajectories must be able to be formed without knowing the arrival times of all parcels to join a platoon beforehand.

Trajectories can be redesigned.
Designed trajectories will, in the real world, most likely not be followed exactly due to external and uncertain factors. In the case of the overflow zone the exact behaviour of a parcel on a conveyor cannot be predicted exactly due to uncertain behaviour like slip. Because of this, the

algorithm must be altered such that trajectories can be redesigned. An obvious event to yield a redesign of trajectories would be the event of a sensor noticing a parcel that is not exactly en route.

Trajectories do not have to contain all parts (I-V).
When designing trajectories, it has been assumed that all parts of the trajectory (I-V) need to be used. This means that the control region is assumed long enough such that vehicles can slow down to the desired minimum velocity and accelerate to the desired maximum velocity. This may not always be the case, especially when redesigning trajectories based on position updates by sensors. In that case, it may happen that a trajectory, for example, is already at part V and only needs to accelerate to reach the endpoint at the desired final velocity. In such a case, it must be possible to design a trajectory that does not consist of all five parts.

Also, it might not be desirable to use each part of the trajectory (I-V) in every situation even though there is enough distance between the parcel and the end of the overflow zone to fit in all parts. It can, for example, happen that a vehicle needs to keep moving at maximum velocity in order to join a platoon that is being formed. If the algorithm does not account for this and the vehicle first needs to slow down, it cannot join the platoon anymore. Therefore, the algorithm needs to be adjusted such that trajectories can be designed that do not consist of all parts (I-V).

All incoming parcels are accepted.
Due to the assumption that parcels do not join the control region if that region is overcrowded, parcels have been rejected to enter the control region if there was not enough space for parcels to decelerate without colliding with other parcels. This means that vehicles that would collide have simply not been considered. For the overflow zone, this implies that the entire system needs to come to a stop if the overflow zone is (too) crowded or that collisions might occur. As this reduces system throughput or safety, a solution must be found such that all incoming parcels can enter the overflow zone without stopping the system.

The length of parcels is taken taken into account when designing trajectories.
Trajectories have been designed such that they do not intersect, as an intersection between trajectories means a collision in real life. However, the length of a vehicle has not been taken into account, meaning that a collision is still possible. As this is undesired in the system and therefore also in the overflow zone, this must be taken into account.

The arrival and departure velocity are different from the maximum velocity.
In the paper, it has been assumed that the arrival and departure velocity are equal to the maximum velocity. This can be seen in Figure 5.1 as the tangent of part I and part V are identical. This does not necessarily have to be the case in the overflow zone, therefore the algorithm needs to be adjusted such that the arrival velocity, departure velocity and maximum velocity can be defined separately.

A minimum velocity bound is defined.
As can be seen in Figure 3.1, part III of the trajectory has a velocity of zero. However, a major problem of the overflow zone at the moment is the start-stop behaviour. Ideally, the trajectories must be designed such that the velocity does not drop below a predefined minimum velocity.

Trajectories are designed to deal with parcels on conveyors that cannot move independently of each other.

The paper considered a situation with autonomous vehicles that could move completely independent of each other. This is different from the situation where parcels are moved by conveyors. The conveyors determine the velocity of parcels and therefore their trajectory to follow. The trajectories, however, are not designed for parcels on conveyors. Therefore, a parcel 'communicates' to a conveyor which velocity it should have. As a single conveyor might carry multiple parcels, it can occur that parcels desire a different velocity for the same conveyor so the desired velocity of a parcel cannot always be satisfied. This will often be the case and cannot be prevented, but taking this into account for the design of trajectories can reduce the desired velocity difference of a conveyor. With that, parcels will follow their planned trajectory better as conveyors can run at the desired velocity more often.

3.6 Contribution

Currently, multiple methods for the design of trajectories to form a platoon at autonomous intersections are presented in literature. These methods are, however, not more than methods to design a trajectory based on predefined arrival and departure times, positions and velocities. When using these methods to actively control autonomous vehicles approaching an intersection in a real-world situation, problems arise. For example:

- A list of future arrival times is needed but not known.
- Trajectories cannot always be followed exactly but alternative trajectories cannot be computed.
- Situations in which there is not enough space for vehicles to find a trajectory without colliding can occur but cannot be dealt with.
- The arrival and departure velocity are equal and identical for all vehicles, while this is not neccesarily the case.

To the best of the writer's knowledge, currently there is no method available in literature that is able to actively control autonomous vehicles approaching an intersection. In the project, current knowledge on platoon forming is used to make a real-time controller that can actively control autonomous vehicles approaching an intersection in a real-world situation. In order to do so, different collections of equations are developed that are used design trajectories from any position. Conditions are introduced that make sure the correct collection of equations is chosen based on based on a parcel's position, velocity and desired departure time.

In addition, in this project platoon forming is considered for parcels on conveyors. This yields some new challenges as the parcels cannot accelerate and decelerate completely independently of each other as the maximum acceleration and deceleration of the conveyor must be taken into account when the gap between parcels becomes small. Furthermore, the conveyors preferably do not stop which introduces a limited time a parcel can be controlled.

In short, the concept of platoon forming in combination with autonomous intersections has been researched before. However, the active control and application domain are, to the best of the writer's knowledge, novel and come with constraints that have not been dealt with before. In the project, current knowledge on platoon forming is used to make a real-time controller for the platoon forming of parcels on conveyors. The performance and influence of design parameters on key performance indicators of this controller are ultimately investigated.

Plant model

4

To verify whether the proposed controller functions as expected, a model-based approach is used to test the control algorithm on a plant model. It is, however, key that the plant model behaves as close to the real world as possible to give useful insight into how the controller would function on a real-world plant. The model is discretized over time where at each timestep model is updated. In this chapter, first the advantages of model-based design are mentioned. Then, it is explained how the plant model works. Important features that determine the behaviour of parcels on the conveyor are discussed in detail in order to validate the accuracy of the model compared to the real world and to verify the implementation of the model in a software environment.

4.1 Model-based design

Model-based design is a model-centric approach to the development of control (Reedy et al., 2010), signal processing (Ahmadian, 2006), communications and dynamic systems (Aarenstrup, 2015). Instead of designing systems by textual specifications and testing software on actual physical systems, in model-based design, a model is used to replace both. The model includes all aspects relevant for the behaviour of the system that is modelled, including the system requirements and constraints, the physical behaviour of system components like actuators and sensors, the algorithm used to control the system and test scenarios to verify the model. Over other designing methods, model-based design comes with some advantages (Bergmann, 2014), (Aarenstrup, 2015) that are described below.

Modular
First of all, the design of a model can be done in a modular fashion. One can start designing on a high-level (system-level) model and then later on improve the accuracy of the model by adding more low-level (component-level) features. Because of this modular design, the complexity of a system becomes easier to grasp. In addition, it becomes possible to test modules of the model separately.

Unambiguous
Models are unambiguous, unlike textual documents. When working with a team on the model, people will generally interpret the model in the same way. This yields clear and efficient communication between people.

Cost-effective
Using model-based design is cost-effective in numerous ways. To start with, models can be used to test a new concept without having to build an expensive physical system. Furthermore, a model can be used to verify if a newly developed algorithm does not contain any errors that cause a system to collide or break down which may lead to high repair costs.

Time-efficient
With the use of model-based design, one is able to implement ideas quickly. Multiple designs can be explored, tested and evaluated in a short time making model-based design a time-efficient approach.

Applicable
When considering a model of a physical system with a control policy, which would be the case in this project, the control algorithm can often be used directly or converted into usable code for the control of the actual physical system. In addition, some model-based design frameworks even allow for automatic code generation based on a model containing requirements and constraints.

In this project, model-based design is used to design the overflow zone. In the model, the plant and the controller are separated. More specifically, virtual sensors are used to detect the presence of a parcel. Then, a control algorithm is developed to determine the desired acceleration of a virtual conveyor. As mentioned in Section 2.4, the control algorithm that controls conveyors is needed to make sure all approaching parcels are able to enter the overflow zone, parcels are slowed down, accumulated and transported in a train to the infeed with minimal stopping. The output of this control algorithm will then be sent to the virtual conveyor that will perform the acceleration. By separating the plant and the controller, the control algorithm uses only sensor information that is also available on a real-world system and outputs only information that can also be used to control real-world conveyors. This results in a control algorithm that can be easily implemented on a real system when desired.

4.2 Modular design and behaviour of conveyors

To start with, a plant model is made that consists of a series of short conveyor belts. Different types of conveyors can be initialized, where each type can have its own size, maximum acceleration and maximum velocity. The plant model can be built with mixed conveyor types and any number of conveyors. In Figure 4.1, an example of a series of short conveyors is shown with conveyor types of different lengths. In the figure, an observe region is visible between x_{observe} and x_{control} and a control region is visible between x_{control} and 0. The use of these regions is explained later, but for now, it is simply used to illustrate different conveyor sizes are possible.

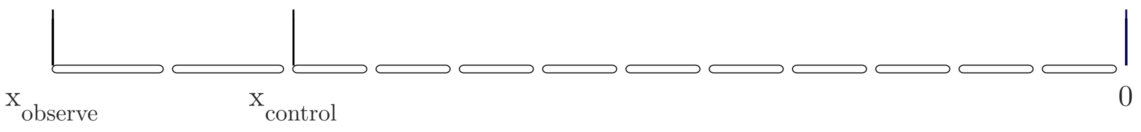


Figure 4.1. Example plant layout with twelve short conveyors.

As mentioned in Section 4.1, the controller and the plant models are separated. To validate that the plant model does not show behaviour that is not possible, in real-life the velocity and acceleration are checked each timestep and compared to the value of one timestep ago. In this way, it can be made sure that the velocity and the acceleration do not exceed any predefined bounds, even if the controller desires differently.

4.3 Behaviour of parcels on conveyors

Next, there is the behaviour of parcels on conveyors. Parcels are given a velocity solely based on the conveyor(s) they touch. This works as follows. First, a randomly chosen centre of mass is given to every arriving parcel. This centre of mass is only determined for a single dimension, namely the direction the parcel moves in. Furthermore, the centre of mass is placed between $\frac{1}{3}$ and $\frac{2}{3}$ of the length of a parcel, as it is unlikely that the centre of mass is near the edge of a parcel. The centre of mass is given once and it does not change after that.

With the centre of mass chosen, a triangle is made between the most downstream bottom point, the most upstream bottom point and the centre of mass at the top of the parcel. Now, the percentage of mass on a conveyor determines the contribution to the velocity of the parcel. With the aid of Figure 4.2, this can be explained.

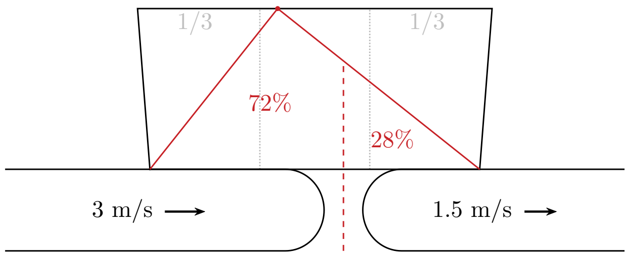


Figure 4.2. Visualization to show the behaviour of parcels on conveyors.

In Figure 4.2, a parcel can be seen on two conveyors. The parcel has a randomly chosen centre of mass between $\frac{1}{3}$ and $\frac{2}{3}$ of the length of a parcel and the conveyors have different velocities. The red dashed line is exactly in the middle of the two conveyors. Numeric values have been given to the conveyor velocities and to the surfaces separated by the red dashed line such that a calculation can be made to show how the parcel velocity is chosen. In this example, 72% of the red triangle surface and therefore the parcel mass is on the upstream conveyor and 28% of the parcel mass is on the downstream conveyor. The upstream conveyor has a velocity of 3 [m/s] and the downstream conveyor has a velocity of 1.5 [m/s]. The parcel velocity v_{parcel} [m/s] is now calculated by

$$\begin{aligned} v_{\text{parcel}} &= 0.72 \cdot 3 + 0.28 \cdot 1.5 \\ v_{\text{parcel}} &= 2.58 \end{aligned}$$

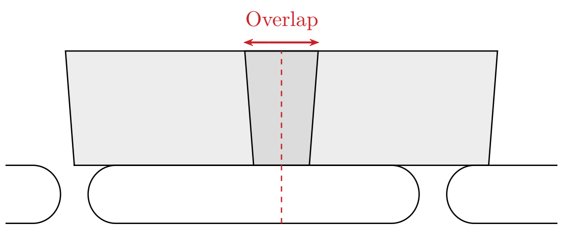
So, in this example, the parcel has a velocity of 2.58 [m/s]. Over time the mass on the downstream conveyor is gradually increased and with that, the velocity is gradually decreased.

This behaviour resembles reality, due to two important aspects. First, there is a realistic transition of parcel velocity between conveyors that are running at different velocities. Instead of a sudden parcel velocity increase or decrease, the parcel velocity gradually changes velocity towards the downstream conveyor velocity. By the time a parcel is completely on a conveyor, the parcel velocity is equal to the conveyor velocity. Second, there is the randomly chosen centre of mass which is different for all parcels. This causes a difference in the exact movement a parcel makes when moving between conveyors that are running at different velocities. As the centre of mass in real-life is also different for each parcel, adding a centre of mass makes the model more realistic.

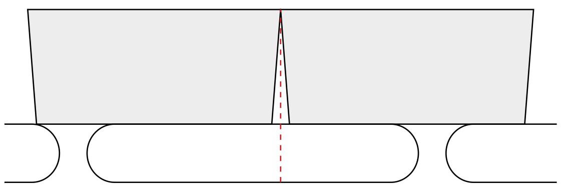
4.4 Behaviour of collisions between parcels

The last feature to discuss is the behaviour of parcels when a collision occurs. A collision is detected when parcels overlap, as overlapping is not possible in the real world. In the plant model, however, an overlap between parcels is possible but must be resolved. Although parcels should collide as less as possible, the behaviour during collisions must be defined.

In case of an overlap between two parcels, the middle point of the overlapping area is found. With the middle point of the overlap found, the two parcels are aligned with this middle point in between the parcels. Essentially, this means that the downstream colliding parcel is pushed forward slightly, and the upstream colliding parcel is slowed down. In Figure 4.3 this method is explained.



(a) Two parcels on conveyors with overlap.



(b) Same parcels on conveyors, aligned with center of overlap.

Figure 4.3. Visualisation of parcel collision behaviour.

This method can only be used in case there are two parcels that collide and both parcels are able to align with the middle point without causing an overlap with other parcels. This is not the case, for example, when a single parcel overlaps with the last parcel of a train. In that case, the closing parcel of a train is not able to move forward without colliding with the consecutive parcel. A way to solve this is by moving the entire platoon forward, however, this is probably not the most realistic behaviour as a single parcel can hardly push a train with a dozen parcels forward. Another way to solve this is by not moving the platoon at all, and assuming the upstream colliding parcel does not affect the platoon. Although this might be true for long platoons, a parcel that collides with a platoon of only two parcels probably affects the position of the platoon.

With this in mind, a decision must be made on how to deal with collisions between parcels, especially when there is not enough space for a collision to be resolved without affecting other parcels. The following is chosen:

> In case a parcel collides with a platoon of two parcels, the platoon moves forward $\frac{1}{3}$ of the overlapping length and the colliding parcel moves back $\frac{2}{3}$ of the overlapping length.
>
> In case a parcel collides with a platoon of three parcels, the platoon moves forward $\frac{1}{4}$ of the overlapping length and the colliding parcel moves back $\frac{3}{4}$ of the overlapping length.
>
> In case a parcel collides with a platoon of four or more parcels, the platoon does not move at all and the colliding parcel joins the platoon.

Formally, this can be described by

$$\begin{aligned}
n_{\text{platoon}} \leq 3 & \begin{cases} \Delta x_{\text{platoon}} & = \frac{1}{n_{\text{platoon}}+1} \cdot l_{\text{overlap}} \\ \Delta x_{\text{parcel}} & = \frac{-n_{\text{platoon}}}{n_{\text{platoon}}+1} \cdot l_{\text{overlap}} \end{cases} \\
n_{\text{platoon}} > 3 & \begin{cases} \Delta x_{\text{platoon}} & = 0 \\ \Delta x_{\text{parcel}} & = -l_{\text{overlap}} \end{cases}
\end{aligned}$$

where n_{platoon} is the number of parcels in the platoon, $\Delta x_{\text{platoon}}$ is the distance the platoon shifts, Δx_{parcel} is the distance the colliding parcels shifts and l_{overlap} is the length of the overlap.

This behaviour resembles the real world well, due to two aspects. First, the magnitude of a collision in real-life is taken into account in the plant model. Instead of approaching all collisions as if they all have the same impact on a parcel's position, a more significant collision results in a larger movement of parcels. This is done by taking the length of the overlap into account when re-positioning colliding parcels. If parcels have a large overlap, the velocity difference between parcels must have been high and the collision must have been relatively big. Hence, the colliding parcels in the overflow zone are moved relatively much. If parcels have a small overlap, the velocity difference between parcels must have been small and the collision must have been relatively small. Hence, the colliding parcels are moved relatively less. Second, the static friction that must be overcome when pushing a parcel in real-life is taken into account by introducing a maximum number of parcels that a single parcel can move. Because of this, a single parcel cannot overcome the static friction of a train longer than three parcels.

4.5 Poisson arrival rate

Besides the behaviour of conveyors and parcels, the arrival rate of parcels at the plant must be discussed. For the arrival rate, a Poisson distribution is assumed, as this allows for a known average arrival rate with a random distribution of arrivals. As it is not known beforehand when parcels will enter the overflow zone, a random (Poisson) distribution is most logical. To verify whether this is implemented correctly, first the properties of a Poisson arrival rate are listed.

1. Events are independent of each other. The occurrence or history of events does not affect the probability of another event occurring.
2. The average (arrival) rate is constant over time.
3. Events cannot occur at the same time.

In the plant model, each timestep a parcel can be generated with the probability of

$$P = \frac{\lambda}{3600} \cdot \delta t \tag{4.1}$$

where P is the probability of an arrival, λ is the arrival rate in [parcels/hour] and δt is the time of a timestep [s] in the simulation. This method is according to the Poisson properties, because:

1. Each timestep, the probability of a parcel being generated is equal. Therefore, events are completely independent of historical events.
2. The probability maintains constant over time and is not affected if the arrival rate in a certain time span is not met.
3. Every timestep, either a single parcel is generated or nothing happens. Therefore, it is not possible to have multiple events at the same time.

In addition, a Poisson arrival rate also allows for trains of parcels to enter the overflow zone, as parcels can be generated in several timesteps close to each other. In this case, parcels initially overlap but are formed into a train as described in the section on the behaviour of collisions between parcels.

Controller

5

In this chapter, the trajectory generation algorithm is explained in detail. First, the proposed trajectory design and why this is the ideal trajectory is explained. Then, the calculations used to construct the trajectories are given.

5.1 Proposed algorithm

Because of all the limitations and additions described above, the algorithm of Timmerman and Boon (2021) must be adjusted to make it work for the overflow zone. Moreover, it is important to discuss what an optimal trajectory looks like and how it relates to the algorithm of Timmerman and Boon (2021).

Define the departure time of the platoon

As stated in Section 2.6, the most important key performance indicator is system throughput. For the overflow zone to contribute to high system throughput, it must deliver platoons of a maximum predefined length to the merge zone. If the platoon is not of maximum length, the density of parcels in the sortation zone is reduced and therefore the system throughput is reduced. To form a platoon of maximum length, enough parcels must arrive in the overflow zone to form the platoon before the platoon leading parcels approaches the merge zone. As stopping is not desired, the platoon leading parcels can only be limited time in the overflow zone with this time mainly determined by the length of the overflow zone and the minimum allowed velocity.

As one does not know beforehand when parcels arrive, or even if enough parcels arrive on time to form a platoon of the desired length, it is key that the platoon leading parcels initially follows the trajectory that takes the most time while taking the boundary conditions into account. The most important boundary conditions for this are the minimum allowed velocity and the desired final velocity. The trajectory that takes the most time and is still feasible starts to decelerate to minimum velocity immediately when entering the control region and accelerates to the desired final velocity as late as possible, as can be seen in Figure 5.1. Initially, this trajectory is given to the platoon leading parcels.

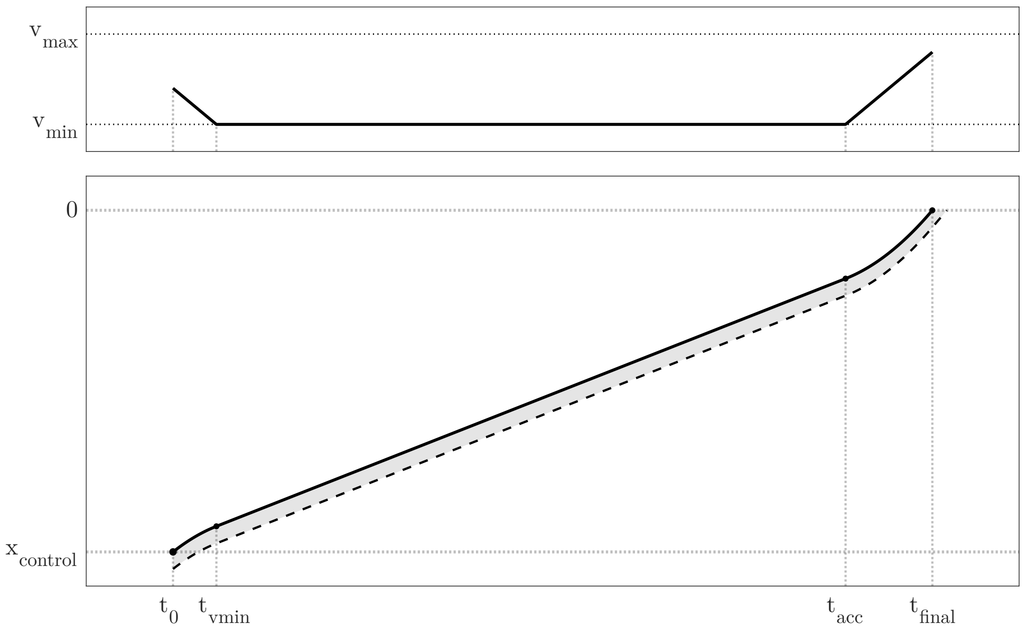
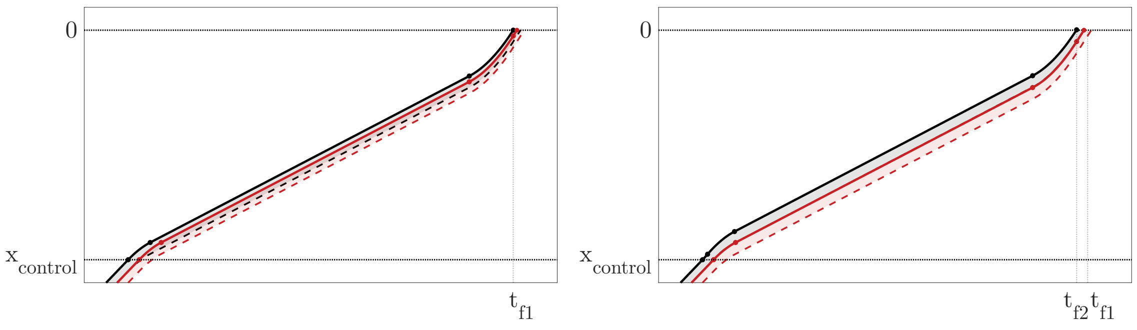


Figure 5.1. Initial trajectory of platoon leader from x_{control} to 0 starting at t_0.

In Figure 5.1, the trajectory of the platoon leader can be seen over time. The top plot shows the velocity over time, the bottom plot shows the trajectory over the same time. The solid line in the trajectory plot represents the trajectory of the head (most downstream point) of the parcels, the dashed line represents the tail (most upstream point) of the parcels. The area between the front and back of the parcels is coloured in light grey and represents the body of the parcels. In the figure, it can be seen that the parcels immediately starts to decrease its velocity at t_0 until the minimum velocity is reached at t_{vmin}. Holding the minimum velocity as long as possible but still leaving the control region with the desired final velocity causes the system to start accelerating at t_{acc}. The point in time at which the parcels crosses the 0 line with the desired final velocity is the departure time of the platoon and is referred to as t_{final}.

With the trajectory designed for the platoon leader, a problem might arise. If there is not enough time between consecutive parcels that arrive at the overflow zone, collisions may occur. This can be explained with the help of Figure 5.2.



(a) Collisions between parcels as the departure time of the platoon leader t_{f1} is set to the initial departure time.

(b) Reevaluated departure time of the platoon leader t_{f2} to avoid collision between parcels.

Figure 5.2. Example of collisions between parcels if two parcels arrive in a train and the departure time is not reevaluated.

In Figure 5.2, two parcels can be seen. The first parcels is the platoon leading parcels and therefore the trajectory that takes the most time is designed. The departure time of the platoon leader, in this case, is called t_{f1}. The second parcels enters just after the first parcels and a trajectory is designed to join the platoon. In Figure 5.2a, it can be seen that the second parcels that enters just after the first parcels, collides with the first parcel as the trajectory intersects with the dashed line of the first parcel, even though the second parcels decelerates with maximum deceleration when entering the control region.

As the second parcels cannot avoid the collision, since stronger deceleration is not possible, the first parcels must adjust its trajectory such that the second parcels can enter without colliding. This can be seen in Figure 5.2b. Here, the second parcels still breaks as hard as possible but the departure time of the first trajectory is adjusted to t_{f2} such that there is no collision and the departure time is still as far in the future as possible. For this to work, the time a parcels approaches the control region must be known just a little time beforehand to make sure the platoon leading parcels does not start decelerating immediately. This is why, besides a *control* region, an *observe* region must be defined.

Adding an observe region in front of the control region
In the observe region, incoming parcels can be observed but not yet controlled. If the observe region is long enough and trajectories can be redesigned and collisions can be prevented. In addition, the control region will not get overcrowded anymore as the departure time is altered such that all parcels can enter and pass through the overflow zone. The minimum length of the observe region is defined by Equation 5.1

$$\begin{aligned} l_{\mathrm{observe}} &\geq v_0 \cdot t_{\mathrm{v0}\rightarrow\mathrm{v_{min}}} \\ l_{\mathrm{observe}} &\geq v_0 \cdot \frac{(v_0 - v_{\mathrm{min}})}{a_{\mathrm{max}}} \end{aligned} \tag{5.1}$$

where l_{observe} is the length of the observe region, v_0 represents the velocity of the parcels in the observe region, v_{min} represents the minimum allowed velocity and a_{max} represents the maximum acceleration and deceleration. If Equation 5.1 holds, any parcels in the control region is able to accelerate from minimum velocity to the velocity in the observe region before the arriving parcels enter the control region. As all parcels in the control region are capable of reaching at least the velocity of an arriving parcels, collisions will not occur.

Define the optimal trajectory
Now the departure time of the platoon leader is defined and an observe region is added to avoid collisions, the optimal trajectory for parcels can be defined.

The general shape of the optimal trajectory closely resembles the trajectory presented by Timmerman and Boon (2021), as shown in Figure 3.1. In Figure 5.3, the general shape of the proposed trajectory can be seen. When comparing it to the trajectory presented by Timmerman and Boon (2021), several differences can be seen.

First, the initial velocity does not have to be equal to the maximum velocity. Therefore, the parcel does not hold its initial velocity until t_{dec} but accelerates to maximum velocity when entering the control region. The reason for this is explained later. This can be seen in part I, where the parcel accelerates with maximum acceleration between t_0 and t_{vmax}. Then, in part II, the parcel holds its maximum velocity until t_{dec}.

Then, the parcel decelerates until t_{vmin}. Contrary to t_{stop} which is defined at Figure 3.1, the parcel does not come to a full stop but follows the minimum defined velocity at part IV. This is done as frequent stopping is undesired for the lifetime of conveyors.

Last, the final velocity does not have to be equal to the maximum velocity. Therefore, the parcels accelerates from t_{acc} until t_{vfinal}, when the desired final velocity is met. In the last part, VI, this velocity is kept.

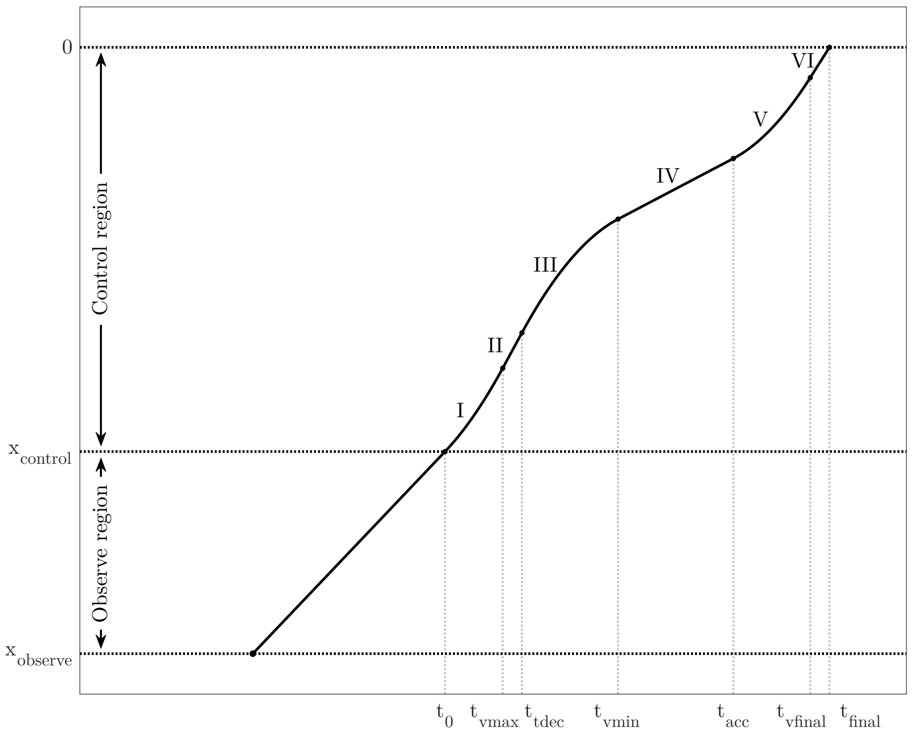
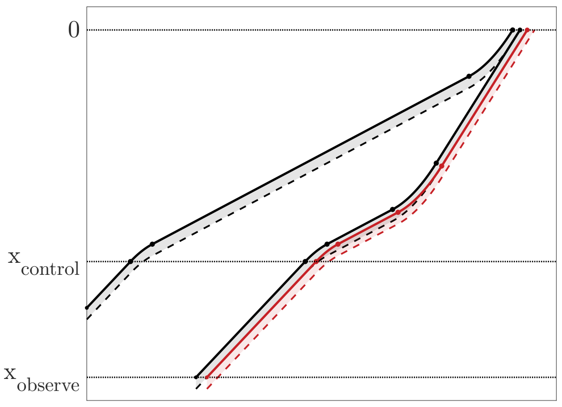
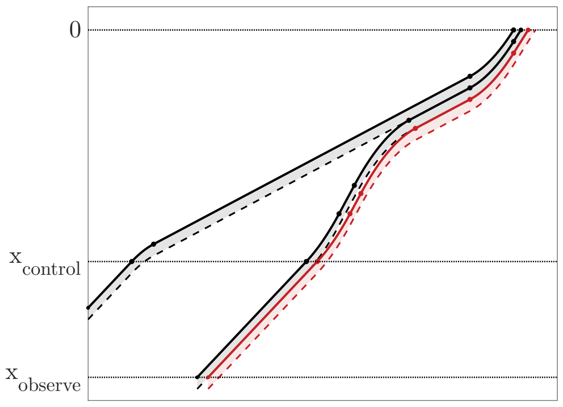


Figure 5.3. General shape of proposed trajectory with labeled parts.

Now the shape of the proposed trajectory has been introduced, it can be explained how this shape helps designing the optimal trajectory. It all comes down to the principle of joining the platoon leader as soon as possible with maximum deceleration. By doing so, as much space as possible is left for other parcels to join without having to advance the departure time of the platoon more than strictly needed. The parcel that follows next, therefore, most of the time has enough space to decelerate to avoid collisions. There is however not always enough space to decelerate. If the overflow zone is too crowded it might happen that there is not enough space to form a train without causing collisions. In this case, the emergency overflow mode as discussed in Section 2.3 is activated to avoid collisions. In order to explain why joining as soon as possible helps avoiding collisions, the opposite of the this trajectory is shown. This is, joining the platoon leader as *late* as possible (Figure 5.4a), instead of as *soon* as possible (Figure 5.4b).



(a) Collision due to immediate deceleration in control region.



(b) No collision with immediate acceleration in control region.

Figure 5.4. Visualization to show why joining the platoon as soon as possible avoids collisions.

In Figure 5.4, the trajectories of three parcels can be seen. The arrival times and departure times of the parcels are the same, but the trajectory in the control region is different. In Figure 5.4a, the second parcel starts decelerating immediately and joins the platoon in the final stage. The third parcel, displayed in red, also decelerates immediately when entering the control region but cannot decelerate fast enough to avoid a collision. This collision is visible as the third parcel crosses the dashed line of the second parcel, comparable to the situation shown in Figure 5.2a. In Figure 5.4b the second and third parcel join the platoon as soon as possible. By doing so and increasing the velocity, the second and third parcel that enter with no gap in between them are separated shortly and no collision occurs. This can be seen in Figure 5.4b, where the distance between the dashed line of the second parcel and the solid line of the third parcel are separated at the beginning of the control region.

Next, the conveyors that transport the parcels must be discussed. Because of the conveyors, not all parcels can be controlled completely individually, as their velocities are determined by conveyors and multiple parcels may be carried by the same conveyor. This can yield to gaps in the platoon and make it difficult for parcels to join the platoon. Therefore, minimizing the instances when there is a difference in the desired conveyor velocity is key.

To cope with this as good as possible, the trajectory is adjusted such that the start of acceleration time t_{acc} and the time final velocity is reached t_{vfinal} is the same for all parcels in the platoon. This is already dealt with, as parcels join the platoon as soon as possible. This can also be seen in Figure 5.4a, where t_{acc} and t_{vfinal} are the same for all parcels where this is not the case in Figure 5.4b. In addition, the instances with a velocity difference that cannot be avoided are made as short as possible. This is explained with the help of Figure 5.5.

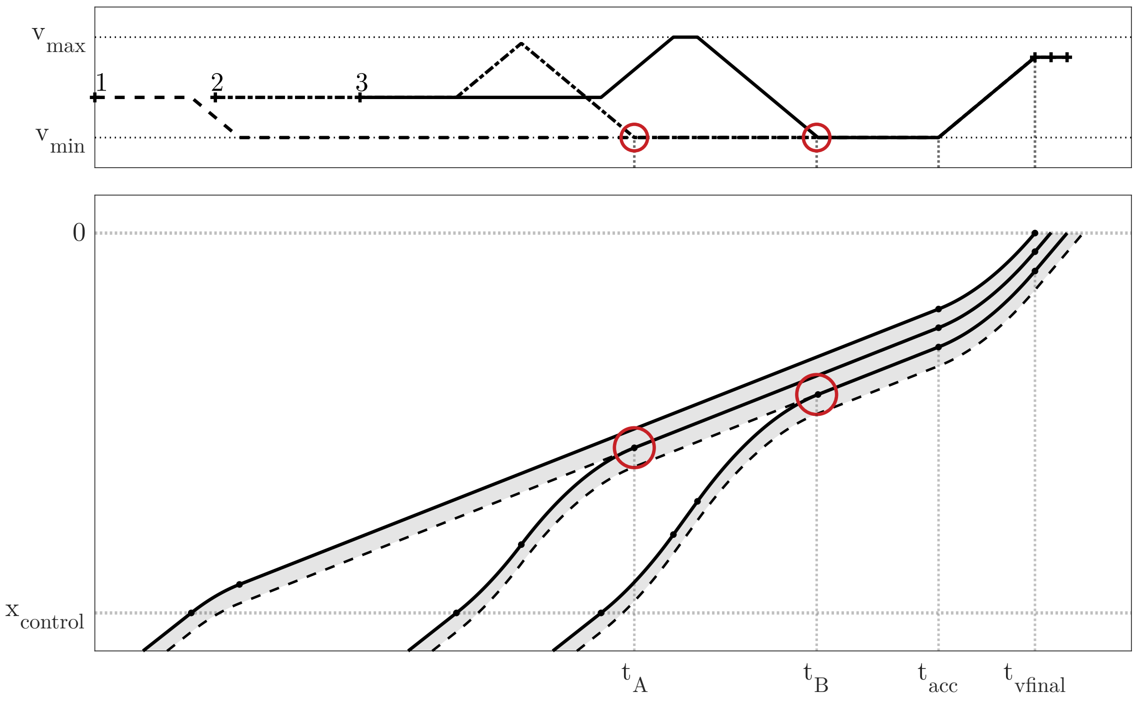


Figure 5.5. Velocity profile of trajectory similar to the trajectory shown in Figure 5.4b

In the top plot of Figure 5.5 the velocity profile over time can be seen for three trajectories. In the bottom plot, the corresponding trajectories can be seen with the same time axis. The first parcel holds its initial velocity until the control region is reached, immediately decelerates to minimum velocity and starts accelerating at t_{acc}. Parcel two arrives at the observe region a bit later and also holds its initial velocity until the control region is met. Then, it starts accelerating but before reaching maximum velocity it starts decelerating, such that it joins the platoon at time t_{A}. Parcels three has a similar trajectory as parcel two but can reach the maximum velocity before joining the platoon. Parcel three joins at time t_{B}.

When taking a closer look at the time t_{A} and t_{B}, in the velocity plot it can be seen that the velocity of parcel two just before t_{A} and the velocity of parcel three just before t_{B} are not equal to the minimum velocity just as the platoon leader. There is, therefore, a difference in velocity just before t_{A} and t_{B}. When looking at the trajectory plot, it can be seen that just before t_{A} parcel two is very close to joining the platoon and just before t_{B} parcel three is very close to joining the platoon. As parcels one and two are close to each other just before t_{A} and parcels two and three are close to each other just before t_{B}, the likelihood of them being on the same conveyor is high. The desired velocity of the conveyors just before t_{A} and t_{B}, however, is different. With that, it can be concluded that there is a difference in the desired conveyor velocity when parcels adjoin a platoon.

Avoiding this difference in the desired velocity is not possible in normal setups, as this requires unrealistic high acceleration and unrealistic short conveyors. To cope with these differences in desired velocities as good as possible, the parcels that join the platoon decelerate with maximum deceleration. In this way, the gap between the platoon and the parcel to join the platoon is decreased as soon as possible. With that, the time the parcels are on the same conveyor with a difference in the desired velocity is minimized.

After the parcels have joined the platoon, it can be seen that the conveyor velocity is the same for all parcels in the platoon. The velocity stays minimal until t_{acc} at which all parcels accelerate with the same velocity. Then, at t_{vfinal}, all parcels hold the same desired final velocity. This can also be seen in the velocity plot in Figure 5.5, where the desired velocity lines of all three parcels lie on top of each other. Because of this, there will not be a difference in the desired velocity of conveyors that causes gaps in the platoon. This results in a platoon with no gaps in between parcels.

From trajectory to conveyor velocity
In case there is a difference in desired velocity, the velocities of the most downstream parcels are taken. When an upstream parcels joins a downstream parcel, the downstream parcel is always the parcel with the lowest velocity. By choosing the velocity of the downstream parcel, the conveyor runs at the lowest desired velocity and with that the departure time is not advanced. Also, slip can be used as an advantage for parcels to join the platoon. By using slip, the parcels to join the platoon can arrive with relatively high velocity on a conveyor with the last platoon parcels with relatively low velocity and slide onto that conveyor to decrease the gap between the parcels and the platoon.

5.2 Trajectory calculations

In Section 5.1, the optimal trajectory and the choices that have been made regarding this trajectory are explained. In this section, the closed-form expressions that are used to design the trajectories are discussed in detail.

Necessity of new equations
The equations used for designing trajectories as presented in (Timmerman and Boon, 2021) do not suffice when taking the proposed adjustments and additions of Section 5.1 into account. First, there is the minimum velocity that is added as well as the initial- and final velocity that do not have to be equal to the maximum velocity. Second, the trajectories must be able to be made from any position with any velocity. This is essential in real-world use on conveyors, as the planned trajectory will most likely not be followed exactly due to slip on conveyors and other external factors. New closed-form expressions must be developed to allow for the proposed changes. As the trajectory must be able to be formed at any time at any position, not every trajectory will consist of all six (I-VI in Figure 5.3) parts anymore. Therefore, a number of trajectory *collections* as well as conditions on when to apply which collection have been defined. A collection consist of the closed-form expressions to calculate the acceleration and deceleration times ($t_{\mathrm{v_{max}}}$, t_{dec}, $t_{\mathrm{v_{min}}}$, t_{acc} and $t_{\mathrm{v_{final}}}$).

Define ideal trajectory
As explained in Section 5.1, the trajectory of the platoon leader is designed such that it spends maximum time in the overflow zone and for the platoon followers the trajectory is designed such that parcels adjoin the platoon as soon as possible. In order to find the closed-form expressions used to calculate the acceleration and deceleration times of the trajectories, first, the lines that define the desired trajectory to join are formulated. In Figure 5.6, the three equations that describe the trajectory of the rear of the platoon leader parcel are visualized.

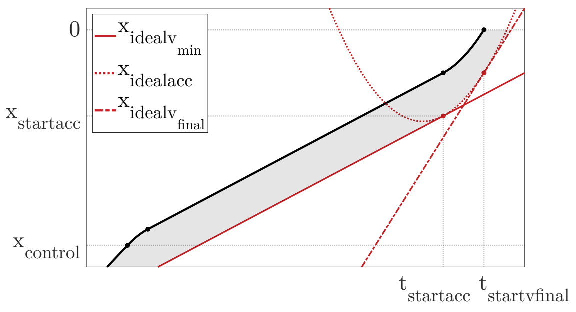


Figure 5.6. Visualization of equations that set up the positions in time to join the platoon.

In Figure 5.6, the trajectory of a single parcel can be seen over time. Taking the length of the parcel, highlighted in grey, into account, the equations that span the most upstream point of the first parcel are determined. This span consists of three parts: the tangent of the minimal velocity part ($x_{\text{idealv}_{\text{min}}}$, IV in Figure 5.3), the acceleration parabola with maximum acceleration (x_{idealacc}, V in Figure 5.3) and the tangent of the final velocity part ($x_{\text{idealv}_{\text{final}}}$, VI in Figure 5.3). The times at which the function that describes $x_{\text{ideal}}(t)$ change are determined by the times at which the tangents and parabola intersect. As the variables t_{acc} and $t_{\text{v}_{\text{final}}}$ are used for each parcel separately and these intersection times are required in the calculations of every parcel, these intersections are given different names: t_{startacc} and $t_{\text{startv}_{\text{final}}}$ as these values define the time at which the platoon leader starts accelerating and reaches final velocity. In Figure 5.6, these times have also been highlighted. The ideal trajectory x_{ideal} [m] over time t [s] is given by

$$x_{\text{ideal}}(t) = \begin{cases} \text{Equation A.1}, & \text{if } t \leq t_{\text{startacc}} \\ \text{Equation A.2}, & \text{if } t_{\text{startacc}} < t < t_{\text{startv}_{\text{final}}} \\ \text{Equation A.3}, & \text{if } t \geq t_{\text{startv}_{\text{final}}} \end{cases}$$

with t_{startacc} [s], the time the platoon leader starts accelerating, defined as

$$t_{\text{startacc}} = t_{\text{tstartv}_{\text{final}}} - t_{\text{v}_{\text{min}} \to \text{v}_{\text{final}}} \tag{5.2}$$

where $t_{\text{tstartv}_{\text{final}}}$ [s] is the time the platoon leaders reaches final velocity and the front of the platoon leader crosses the position is 0 line and $t_{\text{v}_{\text{min}} \to \text{v}_{\text{final}}}$ [s] is the time needed to accelerate from minimum velocity to final velocity defined as

$$t_{\text{v}_{\text{min}} \to \text{v}_{\text{final}}} = \frac{v_{\text{final}} - v_{\text{min}}}{a_{\text{max}}} \tag{5.3}$$

where v_{min} is the minimum allowed velocity [m/s], v_{final} is the desired final velocity [m/s] and a_{max} is the maximum conveyor acceleration and deceleration [m/s^2].

Derive the number of collections needed

Now, it is established that the ideal trajectory consists of three parts. Based on the position and velocity of a parcel, a parcel adjoins the ideal trajectory at one of the three parts. As the closed-form expressions for the trajectory to the ideal trajectory differ based on the part to join, at least three collections of closed-form expressions must be developed. There is, however, another factor that determines what the trajectory should look like. This is, if the maximum velocity v_{max} [m/s] can be reached before adjoining the ideal trajectory. If so, a maximum velocity part (II in Figure 5.3) is present before adjoining the ideal trajectory, else an s-curved trajectory is formed. All combined, a total of six collections must be defined. In Table 5.1, the six collections and what separates them can be seen.

Table 5.1. Decision table for closed-form expression collection.

	Adjoin at v_{min} part	Adjoin at acceleration part	Adjoin at v_{final} part
Able to reach v_{max}	1	2	3
Not able to reach v_{max}	4	5	6

Conditions for collection 1-3

The first three collections are used when maximum velocity can be reached. Based on the position of reaching maximum velocity, the trajectory is chosen. Before going into detail into the calculations, the position and time a parcel is in the controllable region are defined as

$$x_{\text{controllable}} = \mathbf{max}[x_{\text{control}}, x_0] \tag{5.4}$$

$$t_{\text{controllable}} = t_0 + \mathbf{max}\left[\frac{x_{\text{control}} - x_0}{v_0}, 0\right] \tag{5.5}$$

where $x_{\text{controllable}}$ is the position of a parcel [m], equal to the current parcel position x_0 [m] if the parcel is in the control region or equal to x_{control} [m] if the parcel is not yet in the control region. The time $t_{\text{controllable}}$ is the current time [s] plus the time needed to arrive at the control region if a parcel is not yet at the control region, with v_0 being the current parcel velocity [m/s] equal to the arrival velocity before entering the control region.

Condition 1: can maximum velocity be reached?

The first condition that must hold for this collection 1-3 is that a parcel must be able to reach maximum velocity before starting to decelerate. Formally, this can be described as

$$x_{\text{reachv}_{\text{max}}} \leq x_{\text{decline}}(t_{\text{reachv}_{\text{max}}}, v_{\text{max}}) \tag{5.6}$$

where $x_{\text{reachv}_{\text{max}}}$ is the position at which maximum velocity can be reached when accelerating immediately [m], $t_{\text{reachv}_{\text{max}}}$ [s] is the time at which maximum velocity can be reached when accelerating immediately [s] and $x_{\text{decline}}(t_{\text{reachv}_{\text{max}}}, v_{\text{max}})$ [m] represents the line at which decelerating from v_{max} results in the parcel joining the ideal trajectory with the corresponding velocity. Furthermore, $x_{\text{reachv}_{\text{max}}}$ and $t_{\text{reachv}_{\text{max}}}$ are defined as

$$x_{\text{reachv}_{\text{max}}} = x_{\text{controllable}} + \tfrac{1}{2} \cdot a_{\text{max}} \cdot t^2_{\text{v}_0 \to \text{v}_{\text{max}}} + v_0 \cdot t_{\text{v}_0 \to \text{v}_{\text{max}}} \tag{5.7}$$

$$t_{\text{reachv}_{\text{max}}} = t_{\text{controllable}} + t_{\text{v}_0 \to \text{v}_{\text{max}}} \tag{5.8}$$

where $t_{\text{v}_0 \to \text{v}_{\text{max}}}$ is time needed to accelerate from current velocity to maximum velocity defined as

$$t_{\text{v}_0 \to \text{v}_{\text{max}}} = \frac{v_{\text{max}} - v_0}{a_{\text{max}}} \tag{5.9}$$

Next, the equation for $x_{\text{decline}}(t_{\text{reachv}_{\text{max}}}, v_{\text{max}})$ can be defined. As mentioned above shortly, if a parcel approaches the positions defined by this line with maximum velocity and decelerates when crossing, the parcel joins its ideal trajectory at the corresponding velocity. So, x_{decline} is not a trajectory to follow but a line that indicates when to start decelerating. As the ideal trajectory consists of three parts, x_{decline} also consists of three parts. In the first part, the trajectory from maximum velocity to the minimum velocity tangent is defined. In the second part, the trajectory from maximum velocity to the acceleration trajectory is defined, and in the third part, the trajectory from maximum velocity to the final velocity tangent is defined. In Figure 5.7, $x_{\text{decline}}(t_{\text{reachv}_{\text{max}}}, v_{\text{max}})$ can be seen.

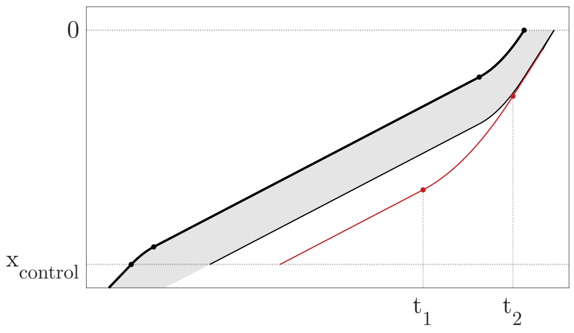


Figure 5.7. Visualization of times that define the equation that is used to span x_{decline}.

To define $x_{\text{decline}}(t_{\text{reachv}_{\max}}, v_{\max})$, first t_1 and t_2 must be calculated. The equations used to define t_1 and t_2 are given in Equation A.5 and Equation A.6 respectively. With these times known, $x_{\text{decline}}(t_{\text{reachv}_{\max}}, v_{\max})$ can be defined.

$$x_{\text{decline}}(t_{\text{reachv}_{\max}}, v_{\max}) = \begin{cases} \text{Equation A.7}, & \text{if } t_{\text{reachv}_{\max}} \leq t_1 \\ \text{Equation A.8}, & \text{if } t_1 < t_{\text{reachv}_{\max}} < t_2 \\ \text{Equation A.9}, & \text{if } t_{\text{reachv}_{\max}} \geq t_2 \end{cases}$$

Condition 2: where to join ideal trajectory if reaching maximum velocity is possible?
If Equation 5.6 is satisfied, a parcel is able to reach maximum velocity before starting to decelerate. Visually, this means that a parcel is able to reach maximum velocity below the red line shown in Figure 5.7 and Figure 5.8.

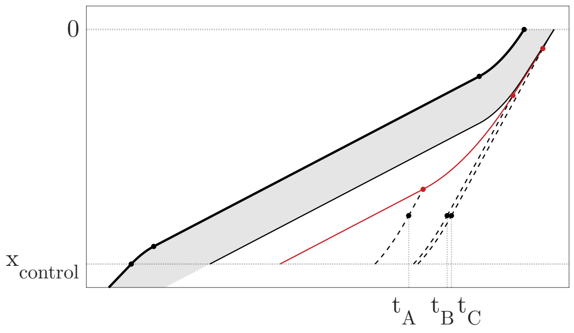


Figure 5.8. Visualization to help with determining which collection that reaches maximum velocity to choose.

Based on the area in which the maximum velocity can be reached when accelerating in the control region as soon as possible, it can be determined at which part of the ideal trajectory the parcel joins the platoon. With that, the correct collection can be chosen.

$$\text{Collection to choose} = \begin{cases} \text{Collection 1}, & \text{if } t_{\text{reachv}_{\max}} \leq t_A \\ \text{Collection 2}, & \text{if } t_A < t_{\text{reachv}_{\max}} \leq t_B \\ \text{Collection 3}, & \text{if } t_B < t_{\text{reachv}_{\max}} \leq t_C \end{cases}$$

The equations used to define t_A, t_B and t_C are given in Equation A.11, Equation A.12 and Equation A.13 respectively.

Collections 1-3
Now, all the conditions that yield the use of the first three collections are stated. Below, the three

collections are shown. As all collections are able to reach maximum velocity, $t_{\mathrm{v_{max}}}$ is the same for all three collections and defined by the time a parcel is in the control region Equation 5.5 plus the time needed to reach maximum velocity can be reached Equation 5.9.

Collection 1: maximum velocity to ideal minimum velocity tangent
To define the times to design a trajectory that first reaches the maximum velocity and then joins the ideal minimum velocity tangent, first, the intersection between the maximum velocity tangent and the ideal minimum velocity is calculated. After that, the time needed to decelerate from maximum to minimum velocity is defined. As the intersection between the two tangents (indicated by the symbol ∩) is exactly in the middle of the deceleration trajectory, the acceleration and deceleration times are defined as

$$
\begin{aligned}
t_{\mathrm{v_{max}}} &= t_{\mathrm{controllable}} + t_{\mathrm{v_0 \to v_{max}}} \\
t_{\mathrm{dec}} &= t_{\mathrm{x_{v_{max}} \cap x_{ideal}}} - \tfrac{1}{2} \cdot t_{\mathrm{v_{v_{max}} \to v_{v_{min}}}} \\
t_{\mathrm{v_{min}}} &= t_{\mathrm{x_{v_{max}} \cap x_{ideal}}} + \tfrac{1}{2} \cdot t_{\mathrm{v_{v_{max}} \to v_{v_{min}}}} \\
t_{\mathrm{acc}} &= t_{\mathrm{startacc}} \\
t_{\mathrm{v_{final}}} &= t_{\mathrm{startv_{final}}}
\end{aligned}
$$

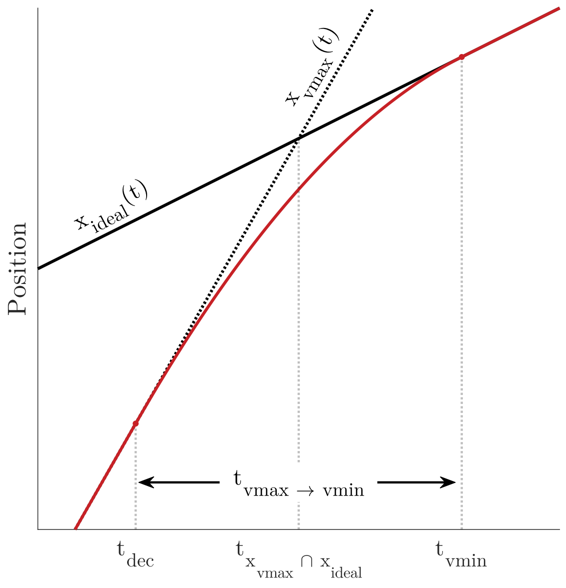


Figure 5.9. Visualization to help with the calculations of collection 1.

In Figure 5.9, a visual aid is given that shows how $t_{\mathrm{x_{v_{max}} \cap x_{ideal}}}$ is constructed and how it helps defining t_{dec} and $t_{\mathrm{v_{min}}}$. Furthermore, the equation used to define $t_{\mathrm{x_{v_{max}} \cap x_{v_{min}}}}$ is defined as Equation B.2. As the parcel joins the ideal trajectory at the minimum velocity part, t_{acc} and $t_{\mathrm{v_{final}}}$ are equal to the acceleration and final velocity time of the platoon leader.

Collection 2: maximum velocity to ideal acceleration trajectory
To define the times of collection 2, first the time the maximum velocity tangent $x_{\mathrm{v_{max}}}(t)$ and the deceleration line $x_{\mathrm{decline}}(t)$ intersect, $t_{\mathrm{x_{v_{max}} \cap x_{decline}}}$, is calculated using Equation B.3. From this time on, the parcel decelerates until $t_{\mathrm{v_{min}}}$. The trajectory between t_{dec} and $t_{\mathrm{v_{min}}}$ is referred to as $x_{\mathrm{dec2ideal}}$ and can be seen in Equation B.4. The time when $x_{\mathrm{dec2ideal}}$ intersects with the ideal trajectory is defined by $t_{\mathrm{x_{dec2ideal} \cap x_{ideal}}}$ and shown in Equation B.6. All combined, this yields to

$$
\begin{aligned}
t_{\mathrm{v_{max}}} &= t_{\mathrm{controllable}} + t_{\mathrm{v_0 \rightarrow v_{max}}} \\
t_{\mathrm{dec}} &= t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{decline}}} \\
t_{\mathrm{v_{min}}} &= t_{\mathrm{x_{dec2ideal}} \cap \mathrm{x_{ideal}}} \\
t_{\mathrm{acc}} &= t_{\mathrm{startacc}} \\
t_{\mathrm{v_{final}}} &= t_{\mathrm{startv_{final}}}
\end{aligned}
$$

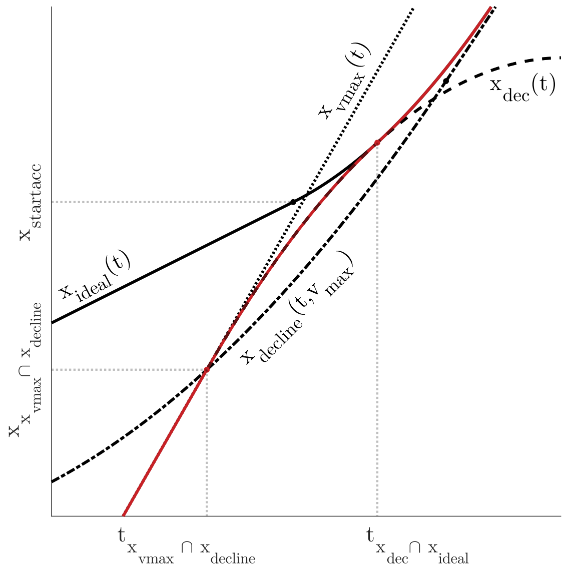


Figure 5.10. Visualization to help with the calculations of collection 2.

In Figure 5.10, a visual aid is given that shows how $t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{decline}}}$ and $t_{\mathrm{x_{dec2ideal}} \cap \mathrm{x_{ideal}}}$ are constructed and how it helps defining t_{dec} and $t_{\mathrm{v_{min}}}$. As the parcel joins the ideal trajectory at the minimum velocity part, t_{acc} and $t_{\mathrm{v_{final}}}$ are equal to the acceleration and final velocity time of the platoon leader.

Collection 3: maximum velocity to ideal final velocity tangent

Collection 3 is essentially the same as collection 1, but in collection 3 the intersection between the maximum velocity tangent and the ideal final velocity tangent must be found instead of the intersection between the maximum velocity tangent and the ideal minimum velocity tangent. The intersection time $t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{v_{v_{final}}}}}$ and the time needed to decelerate from maximum velocity to final velocity $t_{\mathrm{v_{v_{max}}} \rightarrow \mathrm{v_{final}}}$ are given by Equation B.7 and Equation B.8 respectively. Combined, this yields the following acceleration and deceleration times.

$$
\begin{aligned}
t_{\mathrm{v_{max}}} &= t_{\mathrm{controllable}} + t_{\mathrm{v_0 \rightarrow v_{max}}} \\
t_{\mathrm{dec}} &= t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{v_{v_{final}}}}} - \frac{1}{2} \cdot t_{\mathrm{v_{v_{max}}} \rightarrow \mathrm{v_{final}}} \\
t_{\mathrm{v_{min}}} &= t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{v_{v_{final}}}}} + \frac{1}{2} \cdot t_{\mathrm{v_{v_{max}}} \rightarrow \mathrm{v_{final}}} \\
t_{\mathrm{acc}} &= t_{\mathrm{v_{min}}} \\
t_{\mathrm{v_{final}}} &= t_{\mathrm{startv_{final}}}
\end{aligned}
$$

Condition 3: where to join the ideal trajectory if reaching maximum velocity is not possible?

If Equation 5.6 is not satisfied, maximum velocity cannot be reached before decelerating to the ideal trajectory. Therefore, an s-curved shape is used to join the ideal trajectory. The conditions for joining with an s-curved shape are a little different compared to the conditions for reaching maximum velocity. As position and time to join the ideal trajectory depend on the current parcel position and velocity, it is not possible to point out in a 2D time-position plot which collection must be used. This is contrary to condition 1-3, which can be visualized using Figure 5.8. Therefore, the calculations of where to join the ideal trajectory at what time are done for collection 4-6 beforehand and used as conditions before actually designing the trajectories. The correct collection can be chosen by

$$\text{Collection to choose} = \begin{cases} \text{Collection 4}, & \text{if } t_{\text{joinv}_{\text{min}}} \leq t_{\text{startacc}} \\ \text{Collection 5}, & \text{if } t_{\text{joinacc}} \leq t_{\text{startv}_{\text{final}}} \text{ and } x_0 \leq x_{\text{startacc}} \\ & \text{and } v_{\text{min}} \leq v_{\text{intersect}} \leq v_{\text{max}} \text{ and not } t_{\text{joinv}_{\text{min}}} \leq t_{\text{startacc}} \\ \text{Collection 6}, & \text{if } t_{\text{joinv}_{\text{final}}} \leq t_{\text{final}} \text{ and not } t_{\text{joinv}_{\text{min}}} \leq t_{\text{startacc}} \text{ and not} \\ & (t_{\text{joinacc}} \leq t_{\text{startv}_{\text{final}}} \text{ and } x_0 \leq x_{\text{startacc}} \\ & \text{and } v_{\text{min}} \leq v_{\text{intersect}} \leq v_{\text{max}}) \end{cases}$$

Here, $t_{\text{joinv}_{\text{min}}}$, t_{joinacc} and $t_{\text{joinv}_{\text{v}_{\text{final}}}}$ are the earliest times at which a parcel can join the three ideal trajectory parts. If a parcel is able to join the ideal minimum velocity tangent before the ideal acceleration trajectory starts, an s-curved trajectory to the ideal minimum velocity tangent is designed. If this is not possible, but it a parcel is able to join the ideal acceleration trajectory before the ideal final velocity tangent starts, an s-curved trajectory to the acceleration trajectory is designed. Lastly, if a parcel is not able to join the ideal trajectory in the minimum velocity tangent and also not in the ideal acceleration tangent but is able to join at the ideal final velocity tangent, an s-curved trajectory to the ideal final velocity tangent is designed. How the variables $t_{\text{joinv}_{\text{min}}}$, t_{joinacc} and $t_{\text{joinv}_{\text{v}_{\text{final}}}}$ are defined can be seen in Equation A.14, Equation A.17 and Equation A.21 respectively. How these equations are derived becomes clear in the explanation of the collections.

Collections 4-6

Collection 4: s-curved trajectory to ideal minimum velocity tangent

To start off, Equation 5.10 is the general equation to calculate the distance a parcel has travelled with constant acceleration over time given an initial velocity and acceleration. When rewriting Equation 5.10 to isolate the time, Equation 5.11 can be found.

$$s(t) = \tfrac{1}{2} \cdot a \cdot t^2 + v \cdot t \tag{5.10}$$

$$t = \frac{-v \pm \sqrt{2 \cdot a \cdot s + v^2}}{a} \tag{5.11}$$

In Equation 5.11 it can be noticed that the quadratic formula is used with the determinant equal to $2 \cdot a \cdot s + v^2$. As the maximum acceleration and distance to cover are positive in case an s-curved trajectory is made and the velocity is not negative, the determinant is always positive. Therefore, two solutions for t_{scurve} can be found. Obviously, only one of the two is the correct one. As the determinant is always positive, adding the determinant gives a positive t and subtracting the determinant yields a negative t. As the time must be positive, the correct solution can be found by adding the determinant. When rewriting Equation 5.11 for an s-curved trajectory of a parcel, Equation 5.12 can be found

$$t_{\text{scurve}} = \frac{-v_0 + \sqrt{a_{\text{max}} \cdot s_{\text{scurve}} + v_0^2}}{a_{\text{max}}} \tag{5.12}$$

where t_{scurve} is the acceleration and the deceleration time [s]. As an s-curve consists of an acceleration and a deceleration part, the total time of the s-curve is $2 \cdot t_{\text{scurve}}$. Here, s_{scurve} can be formulated as shown in Equation B.9. As can be seen in Equation B.9, the distance that the s-curved trajectory must cover depends on the time the s-curved trajectory takes t_{scurve}. This is, because the parcel adjoins the minimum velocity tangent, which causes the distance to cover to increase over time. This yields some complications, as the time the s-curved trajectory must take depends on the distance to cover, and the distance to cover depends on the time the s-curved trajectory takes. Luckily, this can be rewritten into a single equation for the t_{scurve} as shown in Equation B.10.

A visual representation of t_{scurve} and s_{scurve} can be seen in Figure 5.11. All combined, the following acceleration and deceleration times can be defined.

$$\begin{aligned}
t_{\text{v}_{\text{max}}} &= t_{\text{controllable}} + t_{\text{scurve}} \\
t_{\text{dec}} &= t_{\text{v}_{\text{max}}} \\
t_{\text{v}_{\text{min}}} &= t_{\text{controllable}} + 2 \cdot t_{\text{scurve}} + t_{\text{v0}\rightarrow\text{v}_{\text{min}}} \\
t_{\text{acc}} &= t_{\text{startacc}} \\
t_{\text{v}_{\text{final}}} &= t_{\text{startv}_{\text{final}}}
\end{aligned}$$

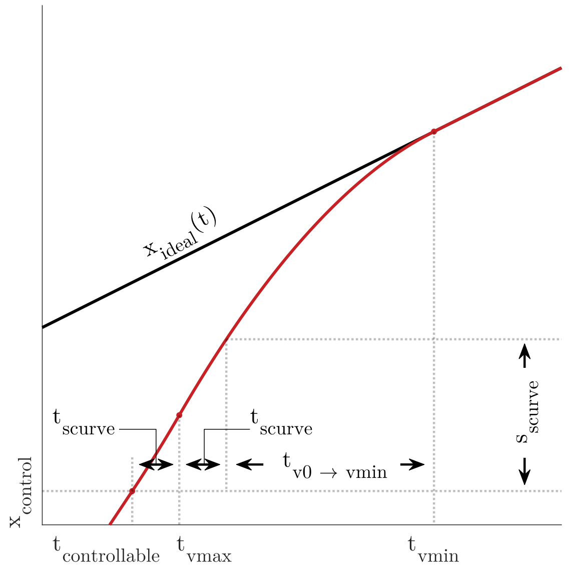


Figure 5.11. Visualization to help with the calculations of collection 4.

As can be seen in the equations above, t_{dec} is equal to $t_{\text{v}_{\text{max}}}$. So, the time that the maximum velocity (or in this case highest possible, as maximum velocity cannot be reached) is reached is equal to the time to start decelerating. In other words, there is no time the parcel spends on maximum velocity. Furthermore, in the equation of $t_{\text{v}_{\text{min}}}$, the time needed to decelerate to minimum velocity is added besides only the time of the s-curved trajectory. This can also be seen in Figure 5.11, where an s-curved trajectory is used to come closer to the ideal minimum velocity tangent but some distance is left for decelerating to the correct velocity to join. As the parcel joins at the ideal minimum velocity tangent, the acceleration time is equal to and t_{startacc} the final velocity is reached at $t_{\text{startv}_{\text{final}}}$.

Collection 5: s-curved trajectory to ideal acceleration trajectory
To formulate an s-curved trajectory to the ideal acceleration trajectory, a more complicated approach is used compared to the s-curved trajectory to the ideal minimum velocity tangent as the ideal trajectory does not have a constant velocity to join anymore. To start with Equation A.8, the line that defines the position to decelerate to join the ideal acceleration trajectory with maximum velocity is used. Now, however, the maximum velocity is replaced with an for now unknown velocity $v_{\text{intersect}}$. This yields to Equation B.11. Next, the first acceleration trajectory of the parcel when entering the control region is defined as Equation B.12.

With Equation B.11 and Equation B.12 defined, the intersection between these two functions can be determined. The time of intersection depends however on velocity $v_{\text{intersect}}$. Therefore, the equation that describes the time of intersection based on $v_{\text{intersect}}$ is defined in Equation B.13. This equation must be equal to the equation that describes the velocity of the first acceleration trajectory over time, given by

$$t_{\text{firstacc}}(v_{\text{intersect}}) = t_{\text{controllable}} + \frac{v_{\text{intersect}} - v_0}{a_{\text{max}}} \tag{5.13}$$

The velocity $v_{\text{intersect}}$ at which Equation B.13 and Equation 5.13 intersect is given by Equation A.15. Now $v_{\text{intersect}}$ is known the intersection time of Equation B.11 and Equation B.12, $t_{\text{x}_{\text{firstacc}}(\text{t})\cap\text{x}_{\text{decline}}(\text{t},\text{v}_{\text{intersect}})}$ can be defined using Equation A.16. With the use of $t_{\text{x}_{\text{firstacc}}(\text{t})\cap\text{x}_{\text{decline}}(\text{t},\text{v}_{\text{intersect}})}$, t_{joinacc} can be defined using Equation A.17. All combined, this yields the following equations

$$\begin{aligned}
t_{\text{v}_{\text{max}}} &= t_{\text{x}_{\text{firstacc}}(\text{t})\cap\text{x}_{\text{decline}}(\text{t},\text{v}_{\text{intersect}})} \\
t_{\text{dec}} &= t_{\text{v}_{\text{max}}} \\
t_{\text{v}_{\text{min}}} &= t_{\text{joinacc}} \\
t_{\text{acc}} &= t_{\text{v}_{\text{min}}} \\
t_{\text{v}_{\text{final}}} &= t_{\text{startv}_{\text{final}}}
\end{aligned}$$

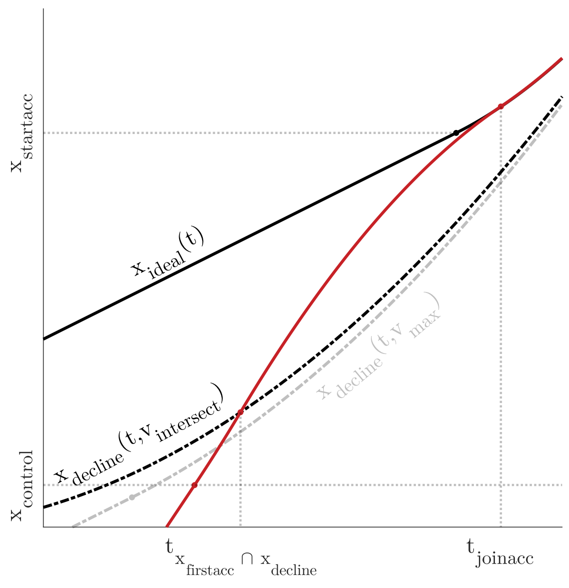


Figure 5.12. Visualization to help with the calculations of collection 5.

As can be seen in the equations, t_{dec} is equal to $t_{\text{v}_{\text{max}}}$ and t_{acc} is equal to $t_{\text{v}_{\text{min}}}$. This means, there is no maximum velocity part and also no minimum velocity part. Last, as the parcel joins at the ideal acceleration part, the final velocity is reached at $t_{\text{startv}_{\text{final}}}$.

One might wonder why to go through the trouble of defining the maximum acceleration parabola, as this is simply the trajectory from the minimum velocity tangent to the final velocity tangent at the latest time instance possible. If the minimum velocity tangent cannot be reached before t_{startacc}, one can immediately join the final velocity tangent and collection 5 and 6 can simply merge, right? Although this trajectory is not wrong as it ends at the right position at the right time with the right velocity, two problems occur. First of all, this is not the optimal trajectory anymore as it does not leave maximum space under the trajectory curve as shown in Figure 5.2. Second, the time the parcel reaches its final velocity $t_{\text{tv}_{\text{final}}}$ is not the same for all parcels anymore. Because of this, the desired overlap in the velocity profile as can be seen in Figure 5.5 after parcels have joined is not present anymore. This yields more differences in the desired velocity which can cause gaps in the platoon at the final stage. Therefore, it is needed to use the maximum acceleration parabola for trajectory designing.

Collection 6: s-curved trajectory to ideal final velocity tangent

The s-curved trajectory to the ideal final velocity tangent works the same as collection 4, where an s-curved trajectory is made to the ideal minimum velocity tangent. Now, s_{scurve} is given by Equation B.15 and t_{scurve} is given by Equation B.14.

$$\begin{aligned} t_{\text{v}_{\text{max}}} &= t_{\text{controllable}} \\ t_{\text{dec}} &= t_{\text{controllable}} \\ t_{\text{v}_{\text{min}}} &= t_{\text{controllable}} + t_{\text{scurve}} \\ t_{\text{acc}} &= t_{\text{v}_{\text{min}}} \\ t_{\text{v}_{\text{final}}} &= t_{\text{control}} + 2 \cdot t_{\text{scurve}} + t_{\text{v}_0 \to \text{v}_{\text{final}}} \end{aligned}$$

Exceptions
In an ideal world, all parcels follow their planned trajectory exactly. In the real world, however, this is not always the case. There are numerous reasons for a parcel to diverge from its planned trajectory. For example, a parcel gets pushed forward by another parcel, the conveyor's velocities deviate slightly from the desired velocity, or slip might cause a parcel to behave differently than planned. Because of this, a parcel might end up at a position with a certain velocity which makes it impossible to end up at the desired position, at the desired time and with the desired velocity. This phenomenon will most likely happen when a parcel is close the leaving the overflow zone, as this leaves little space to correct any disturbances. An intuitive example might be the following:

Imagine a platoon leading parcel that is accelerating to final velocity to leave the overflow zone. The acceleration trajectory is designed such that the platoon leader accelerates as late as possible such that the desired final velocity is reached exactly when leaving the overflow zone, as can be seen in Figure 5.1. If the last conveyor, for example, runs at a slightly lower velocity than expected, the desired final velocity cannot be reached anymore and the parcel leaves the overflow zone not with the desired velocity. Another example can be a consecutive parcel that collides with the platoon leader parcel. This might increase the velocity of the platoon leader and cause the platoon leader to leave the overflow zone not with the desired velocity.

If there is no trajectory that can be formed that leaves at the desired time with the desired velocity, the conditions stated in this section prevent the controller from choosing any collection. There must however be designed a trajectory to control the conveyors, therefore a closer look must be given to the situations in which a correct trajectory cannot be formed. As stated above, this phenomenon will most likely happen when a parcel is close to leaving the overflow zone as this leaves little space to correct any disturbances. Why this is the case is explained in the next paragraph.

With the aid of Figure 5.1, it is explained what happens when no correct trajectory can be designed. As explained in Section 5.1, t_{final} is chosen such that the platoon leader accelerates as late as possible. If a platoon leader is following the minimum velocity tangent and accidentally comes above it, t_{final} is rescheduled such that the parcel is exactly on the ideal minimum velocity tangent again. Therefore, it is not possible for a parcel to come above the minimum velocity tangent. This also holds for platoon followers, as they will collide with the platoon leader instead of crossing the minimum velocity tangent. Problems, therefore, occur in the acceleration and final velocity trajectory. To cope with this, the following has been implemented.

If a platoon leader cannot make a correct trajectory, it is close to leaving the overflow zone. Therefore, the parcel simply accelerates if no correct trajectory can be designed. For platoon followers, a parcel will also be near the acceleration and final velocity trajectory. Collection 6 is used to design a trajectory, as this collection tries to adjoin the final velocity tangent as soon as possible. Although this will most likely not succeed, the parcel tries to leave the overflow zone with the desired velocity at the desired time as good as possible.

5.3 Supervisor

So far only the control of *intra*platooning is considered, meaning the control of multiple parcels within a single platoon. In this section, *inter*platooning is considered, which focuses on the interaction between multiple platoons. As the interplatooning controller is on a higher level (platoon level) than the controller that designs the trajectories (parcel level), the interplatooning is called the supervisor. The supervisor has two main jobs, which are explained below.

Decide if a parcel joins a current platoon or starts its own

When a parcel arrives in the observe region, a decision must be made on whether the parcel joins the last current platoon or if the parcel starts its own platoon. If there is no platoon in the overflow zone (yet), it is a simple choice: a new platoon is formed. However, if there is already a platoon, the decision between joining or starting a new platoon is based on three aspects:

Current number of parcels in platoon

If the current platoon is already of the desired number of parcels, the parcel ideally makes a new platoon. If not, the parcel ideally joins the platoon.

Can a parcel join the platoon

Now it is decided whether a parcel *wants* to join, it must also be investigated if it is possible to join. A calculation is performed to determine if a parcel is able to join based on the latest departure time of that platoon. If a parcel can join and wants to join, it joins. If a parcel can join but does not want to join, if a platoon is already of desired length, it ideally does not join.

Must a parcel join the platoon

Once we know if a parcel *wants* to and *can* join, it must be evaluated if a parcel is obliged to join. A calculation is performed to determine if a parcel is able to be controlled in the control region without interfering with the desired conveyor velocity of the platoon closer of the platoon ahead. If this is not the case, the parcel must join the platoon. Otherwise, if the parcel wants to join and is able to join, it will join the current platoon.

Visually, the decision if a parcel joins a current platoon or if it starts its own yields to the flow chart as shown in Figure 5.13.

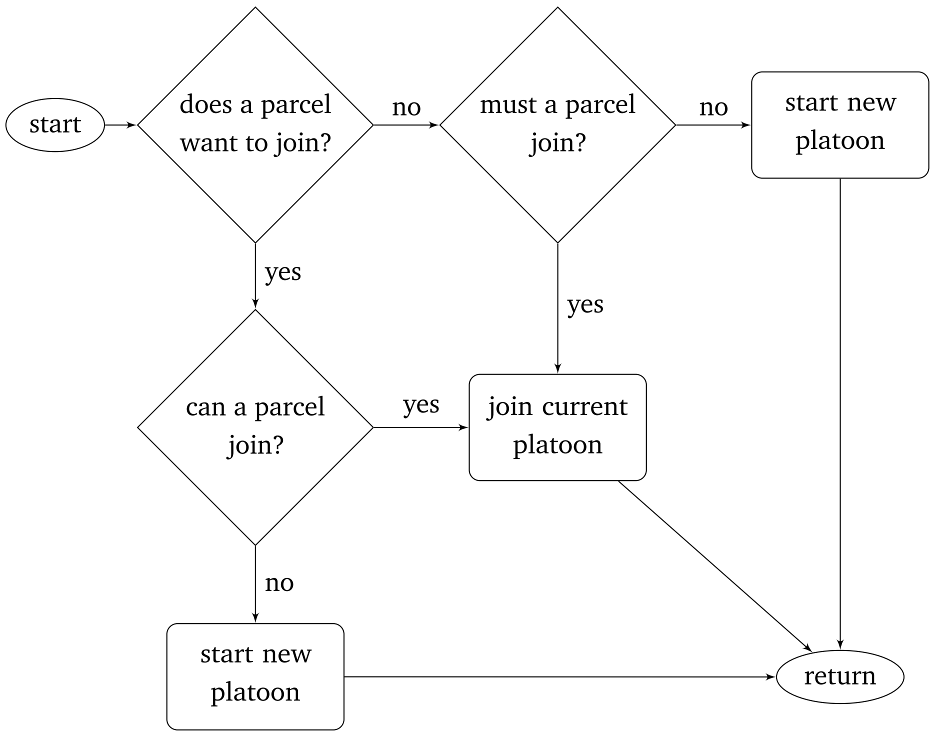


Figure 5.13. Flow chart to decide if a parcel joins a platoon or starts its own.

Define the departure time of the platoon

The second job of a supervisor is to define the departure time of platoons. As explained in Section 5.1, the departure time of a platoon is defined such that the platoon leader spends maximum time in the overflow zone without violating the minimum velocity constraint whilst avoiding collisions with arriving parcels.

To obtain an impression of how the supervisor and trajectory design controller work together to form trains, Figure 5.14 is shown. With the aid of this figure, it can also be explained why the supervisor and controller yield the desired behaviour.

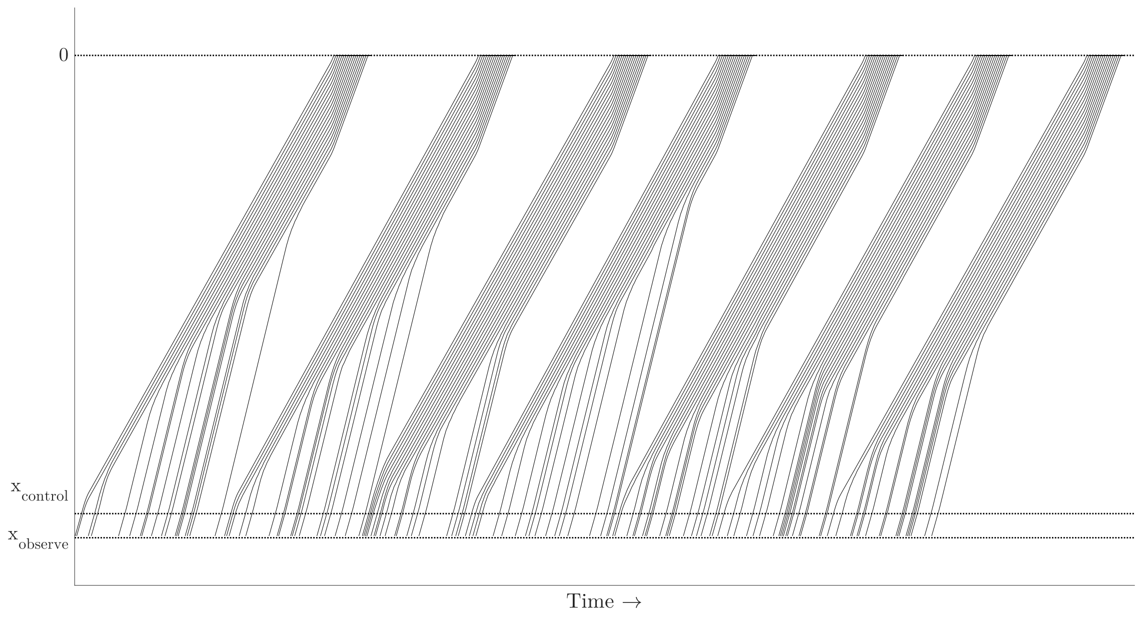


Figure 5.14. Trajectories of parcels formed into platoons over time.

In Figure 5.14, the trajectories of parcels that form multiple platoons can be seen over time. Parcels arrive according to a Poisson arrival rate, as can be seen by the random and 'messy' arrivals in the observe region. At the end of the overflow zone, on the other hand, the neatly formed platoons can clearly be seen and distinguished. Because of these trains, it becomes easy in the merge zone to fit trains from other infeeds between the trains formed in the overflow zone. When comparing this to the arrivals at the observe region, it can be seen that it would be more difficult for parcels from other infeeds to join in between the parcels of the overflow zone. This is due to the close distance between the arriving parcels in the overflow zone, especially when a safety margin must be taken into account.

These observations are directly related to the key performance indicator of having a high system throughput, small footprint and reduction of start-stop behaviour. As none of the parcels comes to a standstill, the start-stop behaviour is completely gone. With the small observe region compare to the control region, the footprint is reduced. Moreover, by the forming of platoons the density on the merge conveyor maintains high and the system capacity does not increase.

In addition, all requirements as mentioned in Section 2.5 are met. The overflow zone is able to handle the capacity as specified in Section 2.5, as well as decelerating and accumulating parcels to form a train. Trains of desired length can be formed and release simultaneously, but unfinished trains can be released as well. No pneumatic parts are used and due to the emergency mode the overflow zone does not overflow itself. Furthermore, the overflow zones is able to handle a burst of parcels and there is no inflow that causes collisions between parcels. Also the constraints as defined in Section 2.5 are taken into account, with a fixed parcel size and a predefined bound on conveyor velocity and acceleration that are not exceeded.

Simulation results 6

A simulation model is made in MATLAB containing to model both the plant and the controller to obtain simulation results. In this chapter, first, a short validation is done to show that the model works as expected. Next, the input and output parameters of the model are discussed in more detail. Last, the influence of design parameters is investigated. Important to note is that in these simulations, full observability is assumed for the controller. This means that on every timestep, the position and velocity of parcels are known.

6.1 Validation

In order to validate if the model works as expected, some simple tests have been performed. To test if the velocity of a parcel solely depends on the conveyor it touches, a parcel is generated and transported over multiple conveyors. One conveyor has a velocity of zero, so it does not run. If the velocity of a parcel indeed depends on the conveyor velocity, the parcel should stop on the conveyor that does not run. When looking at a visualization of the simulation, it can be observed that this is the case.

Second, it is tested if parcels can overlap. If the model works properly, parcels that overlap are separated. This is tested by generating two parcels that overlap. After a single timestep, the parcels should not overlap anymore but align. Again, when looking at the visualization of the simulation, it can be observed that the parcels do not overlap anymore.

Third, the velocity and acceleration of conveyors are logged during a simulation and checked afterwards. The velocity and acceleration should not be outside the defined bounds for the simulation to work properly. Indeed, the test results show that the bounds are not exceeded during simulations.

Fourth, parcels should not be noticed or influence decisions before they enter the observe region. To check if this is true, parcels are generated in front of the observe region. If the parcels are visible, the trajectories do not have to be rescheduled when the parcels enter the overflow zone as their arrival is already expected. If the parcels are not visible, which should be the case, some trajectories of parcels in the control region should change when parcels enter the observe region. This is indeed the case, so it can be concluded that the observe region works as expected.

Fifth, to test if the observe region is long enough, a parcel with minimum velocity at the beginning of the control region is generated and a parcel that arrives with high arrival velocity in the observe region is generated. If the observe region is long enough, the parcels should not collide. This is the case so the observe region is long enough.

Last, the different collections that are used to design the trajectories are tested. For every collection, a parcel with a certain position, velocity and departure time that satisfies all conditions to choose that collection is generated. The trajectory that is generated is evaluated. The parcel should be at the end of the overflow zone at the desired time with the desired velocity. If this is not the case, the trajectory is not designed correctly. For all collections, a trajectory is checked and all collections are correct.

6.2 Output parameters

Before results can be obtained from the simulation model, the output parameters must be defined. As mentioned in Section 2.6, the most important key performance indicator is the system throughput. For the overflow zone, the platoon length that is departed to the merge zone has an influence on the system throughput as windows of fixed size are reserved in the merge zone. If a platoon is of desired length, a higher density on the merge zone is reached which results in higher system throughput. Therefore, the platoon length is the output parameter that is looked at. The platoon length can be defined in either the number of parcels in a train or by the total length of a platoon. It is chosen to proceed with the latter, as the total length of the platoon also takes gaps in the platoon into account.

In Section 2.6, two other key performance indicators are mentioned. First, there is the desire to decrease the footprint. Second, there is the desire to decrease power consumption, which directly relates to conveyor properties like maximum velocity and acceleration. Simulations are therefore done with different parameters to find out how the footprint and power consumption affect the ability to form a platoon. The output parameter (the train length) is used to find relations between parameters. The exact value of the output is not that relevant.

6.3 Input parameters

When running a simulation, input parameters are required. These input parameters consist of system parameters on which the overflow zone has no influence, design parameters of the overflow zone, control choices for the platooning algorithm and simulation settings. In this section, the input parameters are discussed. First, in Table 6.1, the input parameters and their corresponding initial value are stated. Then, below the table, all parameters are discussed in more detail.

Table 6.1. Initial input parameters with chosen values for simulation.

Input parameter	Value	Unit	Type of parameter
Arrival rate	2000, 5000, 8000	[parcels/hour]	System parameter
Arrival velocity	3.0	[m/s]	System parameter
Departure velocity	2.31	[m/s]	System parameter
Parcel length	0.458	[m]	System parameter
Desired platoon length	18	[-]	System parameter
Number of conveyors	60	[-]	Design parameter
Conveyor length	0.6	[m]	Design parameter
Maximum conveyor velocity	3.5	[m/s]	Design parameter
Maximum conveyor acceleration	2.5	[m/s^2]	Design parameter
Minimum conveyor velocity	1.5	[m/s]	Control parameter
Simulation sample time	0.05	[s]	Simulation parameter
Simulation time	30	[min]	Simulation parameter

Arrival rate

The arrival rate defines the number of parcels that arrive at the overflow zone over time. As discussed in Chapter 4, the arrival rate follows a Poisson distribution. To see how the overflow zone performs at different arrival rates, all simulations are run with a low, medium and high arrival rate. For the low arrival rate, a total of 2000 parcels per hour is chosen. As the required arrival rate equals 10% of the system throughput of 24000 parcels per hour which is 2400 parcels per hour, 2000 parcels per hour is a realistic value for a low arrival rate. For the medium arrival rate, approximately 20% of the system throughput is chosen at 5000 parcels per hour. Lastly, for high arrival rate, approximately 30% of system throughput is chosen at a rate of 8000 parcels per hour. Initially, a medium arrival rate is chosen, as this is conform to the requirement but is not excessively high.

Arrival velocity
The arrival velocity defines the velocity at which parcels arrive at the overflow zone and are transported in the observable region. The arrival velocity differs slightly in the Posisorters of Vanderlande and is equal to the velocity of the sortation zone. This is generally a relatively high velocity of about 3 [m/s]. Therefore, a value of 3 [m/s] is chosen.

Departure velocity
Like the arrival velocity, the departure velocity defines the velocity at which parcels are to leave the overflow zone. The departure velocity is generally slightly lower than the arrival velocity of the overflow zone and is based on the velocity at the merge zone. As the merge conveyor has a velocity of approximately 2 [m/s] and the angle at which the overflow zone attaches to the merge zone is 30 degrees, a departure velocity is chosen of $2/\cos(30°) \approx 2.31$ meter per second.

Parcel length
The parcels that enter the overflow zone are all placed in a tray or tote, as defined in Section 2.2. As the overflow zones in current Posisorters of Vanderlande only use a *homogeneous flow*, it is chosen to only use trays for the simulation. As can be seen in Section 2.2, the length of a tray equals 0.458 meters.

Desired platoon length
The desired platoon length defines the ideal length of a platoon. As the Posisorters of Vanderlande make trains of eighteen now, the merge zone consists of windows that fit a train of eighteen parcels. Therefore, it is chosen that the desired platoon length also equals eighteen.

Number of conveyors
The number of conveyors defines, as the name suggests, the number of conveyors that are used during the simulation. This includes all conveyors, so both the conveyors in the *control region* and the *observable region*. In combination with the conveyor length, the length of the overflow zone is chosen. An overflow zone can take any length up to several hundreds of meters, therefore there is not a single number of conveyors or overflow length that is recommended for simulation. As Vanderlande desires to reduce the footprint, it is chosen to keep the overflow zone relatively short by using 20 to 100 short conveyors.

Conveyor length
Besides the number of conveyors, also the length of the conveyors is defined. It is chosen to use the same conveyor length for all conveyors in the simulations, as it makes sense that the overflow zone

in real-life is made out of a series of identical conveyors. Vanderlande currently has conveyors of 0.6 and 0.75 meters. Therefore, an initial value of 0.6 meters seems realistic.

Maximum conveyor velocity
The maximum conveyor velocity defines the maximum velocity at which a conveyor can run. The maximum conveyor velocity differs between conveyor types, but high-speed conveyors have a maximum velocity of around 4 [m/s]. Using an initial maximum velocity of 3.5 [m/s], therefore, seems realistic.

Maximum conveyor acceleration
The maximum conveyor acceleration defines the maximum acceleration at which a conveyor can accelerate and decelerate. Current conveyors of Vanderlande have maximum accelerations of approximately 2-3 [m/s^2], therefore 2.5 [m/s^2] is chosen to be a realistic initial value.

Minimum conveyor velocity
The minimum conveyor velocity does not determine the minimum velocity at which a conveyor can physically run, but the lower bound for the conveyor velocity that is defined in the control algorithm. In essence, it defines the minimum velocity with which the platoon moves. As moving with a velocity approaching zero contradicts the idea of platoon forming, it is chosen not to go below 0.5 [m/s]. The initial value is set at 1.5 [m/s], which is still slower than the arrival and departure velocity.

Simulation time
Before being able to run a simulation, the simulation time must be defined. The simulation time defines the time the model runs in the simulation world. The simulation time must be long enough to ensure the results coming out of the simulation have converged and any start-up effect is negligible such that the outcome is reliable. A longer simulation time does not influence the results significantly, but is time consuming and therefore not desired.

With all the system, design and control parameters defined, the simulation and sample time can be determined. Unfortunately, one cannot be determined without the other. Therefore, an initial guess of one of the two must be made to start the simulations. It is chosen to take an initial sample time of 0.05 seconds. If this sample time is small enough is investigated in the next section. With all the input parameters defined, the simulation time can be determined.

In order to determine the simulation time, two steps are needed. First, the value to which the simulations converge must be determined. Second, a value must be given to how well simulations converge over time. To start with, 25 simulations with a length of 60 minutes are run. All simulations combined, over 8000 trains have been formed. With the length of each train known, the average length of a train can be determined within a certain confidence interval. This is done in the following steps:

First, the mean of the train length $\bar{x}$ in [m] is calculated using

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \tag{6.1}$$

where x is the set of all train lengths in [m] and n is the total number of formed trains [-]. Second, the standard deviation σ [-] of this data set is determined using

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}} \tag{6.2}$$

With the mean $\bar{x}$ and standard deviation σ known, the confidence interval can be determined. First, a confidence level must be chosen. A common confidence level is 95%, so that number is also what is used here. The error ϵ in [m] is defined as

$$\epsilon = Z_{\frac{\alpha}{2}} \cdot \frac{\sigma}{\sqrt{n}} \tag{6.3}$$

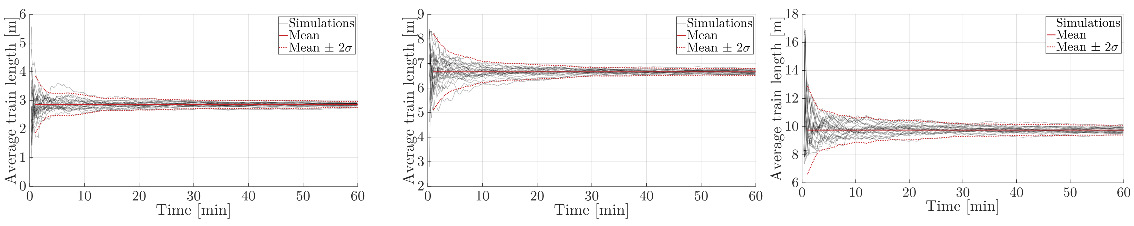
with Z the confidence coefficient [-] and α the confidence level of 95%. The confidence coefficient Z consists of a value that can be found in a *z table* based on the confidence level. A confidence level of 95% corresponds with a confidence coefficient of approximately 1.96. Last, the confidence interval is given by

$$\bar{x} \pm \epsilon \tag{6.4}$$

When applying the above equations for the simulation data with an arrival rate of 5000 [parcels/hour], it can be found that $\bar{x}$ = 6.66 [m] and ϵ = 0.03 [m]. This means, with 95% certainty, it can be said that the mean train length is equal to 6.66 meters with an error of ± 0.03 meters. For an arrival rate of 2000 [parcels/hour] it comes down to 2.86 ± 0.02 [m] and for 8000 [parcels/hour] it comes down to 9.76 ± 0.07 [m].

Now the average train length to which the simulations must converge is known, a value must be given to how well simulations converge to this average train length over time. As mentioned, for determining the mean of the train length for the current simulation settings, all trains formed during 25 simulations are considered as separate data points ($n >$ 8000). For determining when a simulation has converged to the calculated mean, however, one should not look at all trains separately. Instead, every timestep the average train length up to that timestep of each simulation separately can be determined. When doing this for every simulation ($n =$ 25), at every timestep the standard deviation can be calculated using Equation 6.2 with x being the average train length [m] up to the current timestep of all 25 simulations.

By definition, the standard deviation states that 68% of the results are between $\pm\sigma$ and 95% of the results are between $\pm 2\sigma$. Therefore, the standard deviation can be used to assign a value to how well simulations converge to the mean over time. In Figure 6.1 the average train length of all 25 simulations can be seen over time, with the mean and 2σ indicated as well. In Table 6.2 the values of 2σ that are indicated in Figure 6.1 are listed for different simulation times.



(a) Arrival rate 2000 [parcels/hour]. **(b)** Arrival rate 5000 [parcels/hour]. **(c)** Arrival rate 8000 [parcels/hour].

Figure 6.1. Average train length over time for 25 simulations with different arrival rates with mean and 2σ shown.

Table 6.2. Standard deviation of 25 simulations for different arrival rates.

Simulation time [min]	2σ [m] for 2000 [parcels/hour]	2σ [m] for 5000 [parcels/hour]	2σ [m] for 8000 [parcels/hour]
5	0.39	0.72	1.31
10	0.33	0.43	0.85
15	0.21	0.33	0.73
20	0.19	0.29	0.69
25	0.18	0.25	0.58
30	**0.15**	**0.17**	**0.44**
35	0.14	0.17	0.43
40	0.13	0.15	0.38
45	0.13	0.14	0.39
50	0.13	0.13	0.37
55	0.12	0.15	0.36
60	0.10	0.14	0.35

As shown in both Figure 6.1 and Table 6.2, the value of the standard deviation reduces significantly in the first 30 minutes but does not reduce much after that time. Therefore, it can be concluded that the simulations have converged at 30 minutes.

Lastly, it must be determined if the simulations after 30 minutes have converged *enough* at 30 minutes. In other words, is the confidence interval in which simulations end after 30 minutes small enough to give good results? When looking at the corresponding standard deviation, it can be seen that 95% of the results are in the interval of the mean $\pm$ 2σ" after a simulating period of 30 minutes, with 2σ being 0.15, 0.17 and 0.44 [m] for an arrival rate of 2000, 5000 and 8000 [parcels/hour] respectively. The mean, however, is not exactly known. Therefore, to determine if the interval is small enough, one more step is required. The probability of the mean being in the calculated interval of ($P(A)$) as well as the average train length being in the calculated interval of two times the standard deviation ($P(B)$) is defined by

$$\begin{aligned} P(A \cap B) &= P(A) \cdot P(B) \\ P(A \cap B) &= 0.95 \cdot 0.95 \\ P(A \cap B) &\approx 0.90 \end{aligned} \tag{6.5}$$

So, when considering the normal arrival rate of 5000 parcels per hour, with just over 90% certainty it can be stated that a simulation run for 30 minutes ends within both intervals. In the worst case, this interval consists of an error of 0.03 meters caused by determining the mean as well as an error introduced by the standard deviation of 0.17 meters. All combined, it can be said with 90% certainty that a simulation run for 30 minutes has an error of $\pm$ 20 centimetres from mean 6.66 [m]. To put this into perspective, this is only 3% of the average train length with the current simulation settings and 2% of a full train length of eighteen trays. For an arrival rate of 2000 [parcels/hour] this comes down to an error of 1% from mean 2.86 [m] and for an arrival rate of 8000 [parcels/hour] this comes down to an error of 5% from mean 9.76 [m]. As these errors are small enough to draw useful conclusions about the influence of input parameters on the output parameter, it is concluded that the simulation has converged enough after 30 minutes of simulation.

Sample time
The second parameter that needs to be chosen is the sample time. As an iterative model is made, the time between each timestep in the simulation must be defined. It is important that the sample time is small enough to ensure the behaviour of the plant resembles reality, but a too small sample time affects the computation time of a simulation in a negative way. Therefore, the sample time must be chosen with care.

To investigate if the sample time is small enough, at first, a simulation of 30 minutes is done with very small timesteps of 0.005 [s]. This means there are 200 steps every second. A simple calculation shows that a parcel is at least 50 timesteps on every conveyor with a sample time of 0.005 [s], which should be enough to model the behaviour correctly. The output of the simulations are the lengths of the trains that are formed. With the same input, the output should also be the same. The results from the simulation with timesteps of 0.005 [s] can now be compared with the simulations with bigger sample times. If the results are similar, the bigger sample time is still small enough to give credible results.

A widely used method to find out if two data sets are identical is the so called *T-test*. With a T-test, a *T-score* can be calculated that says something about the difference *between* and the difference *within* two data sets. The equation used to calculate the T-score is given in Equation 6.6

$$T = \frac{\frac{\sum_{i=1}^{i=N}(x_i - y_i)}{N}}{\sqrt{\frac{(\sum_{i=1}^{i=N}(x_i-y_i))^2 - \frac{\sum_{i=1}^{i=N}(x_i-y_i))^2}{N}}{(N-1)N}}} \tag{6.6}$$

where N is the number of data points and x and y are the data points of the sets to compare. The T-score can, with the use of a lookup table, be converted into a *P-value*. The P-value is a value between 0 and 1 that is used to support or reject a null hypothesis. In this case, the null hypothesis is that the results of the two simulations with different sample times are the same. Generally, if a P-value is below 0.05, the null hypothesis can be rejected. In other words, the simulations do not yield the same results which says that the sample time is chosen too big. If a P-value is above 0.05, the null hypothesis cannot be rejected so the chosen sample time gives credible results.

This T-test is performed for a sample time of 0.01 [s] and 0.05[s]. The resulting P-values are 0.46 and 0.51 respectively. As these numbers are way above 0.05, it can be concluded that the null hypothesis cannot be rejected and with that it cannot be said that the results with different sample times differ. Therefore, a sample time of 0.05 is used.

6.4 Real-time control

An important reason for designing trajectories with based on acceleration and deceleration times is the computational efficiency. As the controller is developed for real-time usage, it should be fast enough. In the Figure 6.2, the computation time of the controller can be seen for 36000 timesteps with the input parameters as shown in Table 6.1.

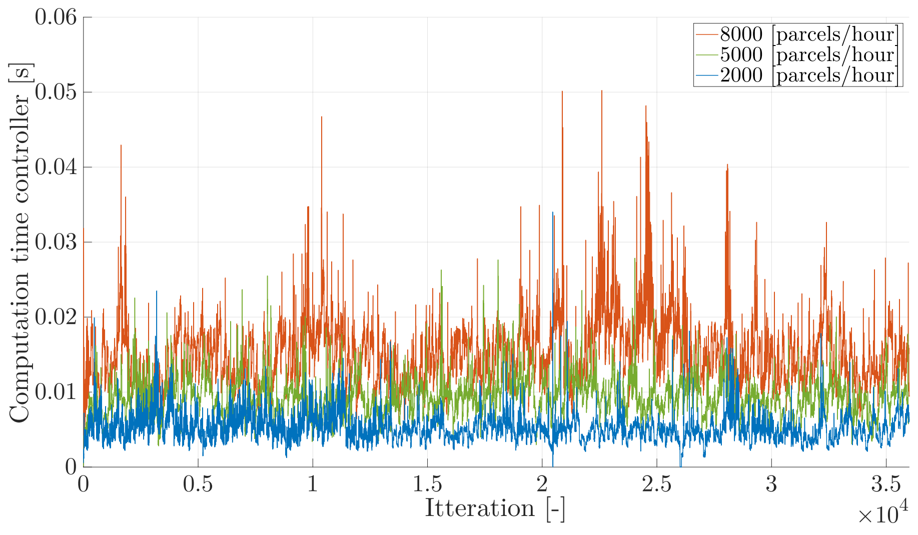


Figure 6.2. Computation time of controller for arrival rate of 2000, 5000 and 8000 [parcels/hour].

From Figure 6.2, two conclusions can be drawn. First, the computation time of the controller increases when the arrival rate increases. This can be explained by the number of parcels that must be controlled at the same time. If the number of parcels increase, more trajectories must be designed and the computation time increases. Second, the mean computation time is of the controller equals 0.005, 0.009 and 0.015 seconds for the arrival times of 2000, 5000 and 8000 [parcels/hour] respectively. In a worst-case scenario, where the arrival rate is excessively high, still over 20 computations can be done in one second. As this is for sure enough for real-time control, it can be concluded that real-time control with this controller is possible.

6.5 Influence of design parameters

The output parameters, the train length, can be displayed in a box plot. For the following input parameters, a box plot is made in order to find the influence of a parameter on the train length. Every parameter is simulated with an arrival rate of 2000, 5000 and 8000 parcels per hour.

Table 6.3. Range of input parameters of which the influence on the KPIs is tested.

Input parameter	**Value**	**Unit**	**Type of parameter**
Arrival rate	2000, 5000, 8000	[parcels/hour]	System parameter
Number of conveyors	20, 40, **60**, 80, 100	[-]	Design parameter
Conveyor length	0.2, 0.4, **0.6**, 0.8, 1.0	[m]	Design parameter
Maximum conveyor velocity	2.5, 3.0, **3.5**, 4.0, 4.5	[m/s]	Design parameter
Maximum conveyor acceleration	1.5, 2.0, **2.5**, 3.0, 3.5	[m/s^2]	Design parameter
Minimum conveyor velocity	0.5, 1.0, **1.5**, 2.0, 2.5	[m/s]	Control parameter

In every box plot, the desired train length is shown with a dashed line. The closer the train length is to this dashed line, the better the results are. Before discussing the results in more detail for each input parameter that is altered, the results with an arrival rate of 8000 must be discussed. As can be seen in most box plots when considering an arrival rate of 8000, many trains are formed that exceed the desired train length. This is due to this high arrival rate that in real-life situations most likely will not happen. These results show that the controller also works for a very high arrival rate, but the results do not say much about the influence of design parameters on the output parameter.

In addition, results with an arrival rate of 2000 parcels per hour often do not reach the desired train length. This is because the arrival rate is too low to form a train of the desired length whilst not violating the minimum velocity constraint. Having a shorter train than desired is not by definition a problem. However, as windows in the merge zone long enough to hold a full train, the density on the merge conveyor is decreased and with that the system throughput. Therefore, for the analysis of the influence, mostly the arrival rate of 5000 parcels per hour is used.

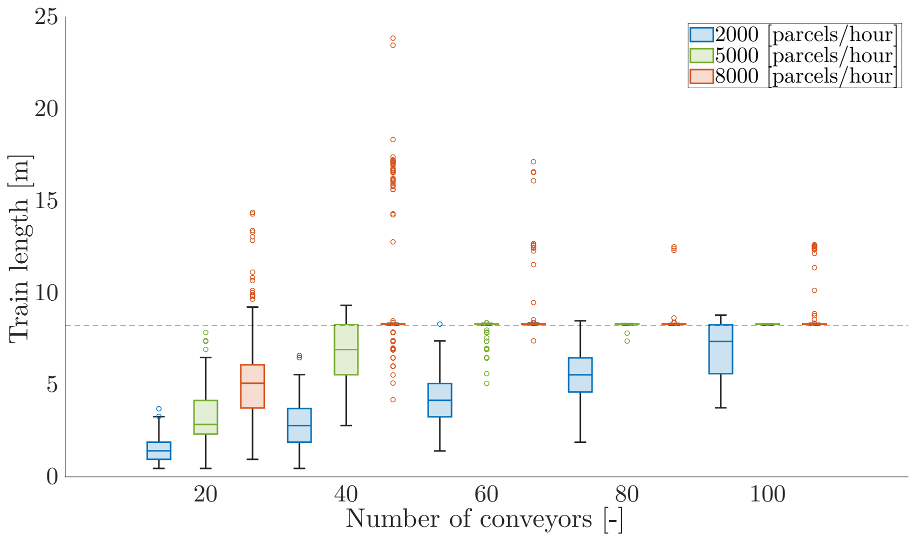


Figure 6.3. Box plot of train length for different number of conveyors with arrival rates of 2000, 5000 and 8000 [parcels/hour].

First, there is the number of conveyors. It can be seen that more conveyors yield a longer train. This makes sense, as more conveyors give a longer overflow zone so there is more time for parcels to join a platoon. Also, in Figure 6.3, it can be seen more conveyors can deal with a high arrival rate better as the number of trains that exceed the desired length becomes less when the number of conveyors increases. This also makes sense, as there is more room to form trains.

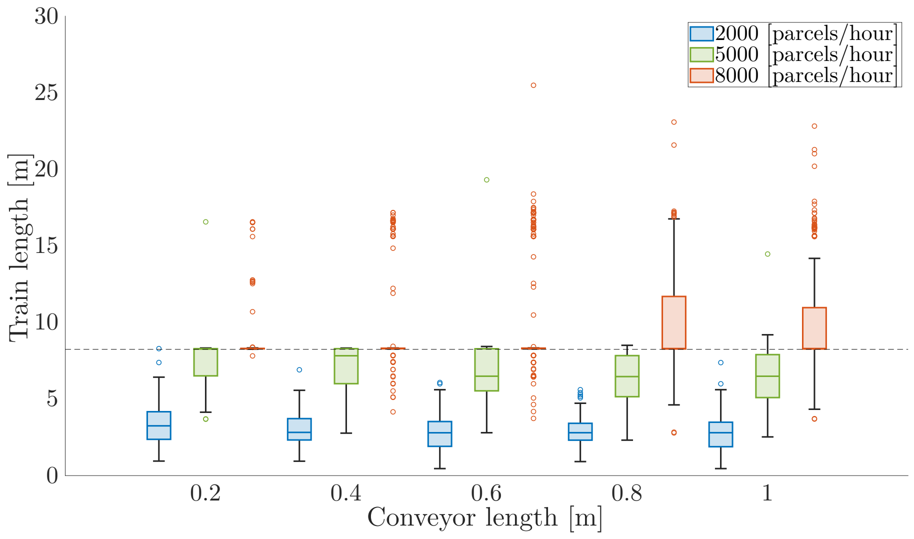


Figure 6.4. Box plot of train length for different conveyor lengths with arrival rates of 2000, 5000 and 8000 [parcels/hour].

Second, the conveyor length is altered. Here, it is made sure that the overflow zone itself keeps the same length, so the number of conveyors drops when the length of conveyors increases. It can be seen that short conveyors perform slightly better, but do not have a big influence. This slight increase of performance can be explained by the fact that shorter conveyors make it easier for parcels to join a platoon as the difference in desired velocity for conveyors becomes less.

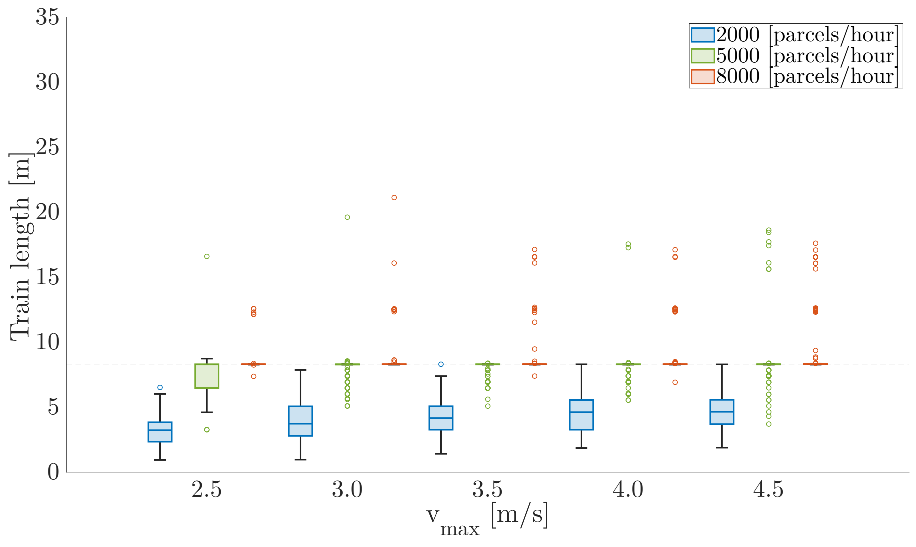


Figure 6.5. Box plot of train length for different maximum velocities with arrival rates of 2000, 5000 and 8000 [parcels/hour].

Third, there is the maximum velocity. It can be seen that a higher maximum conveyor velocity yields slightly better results. This makes sense, as a higher maximum velocity allows parcels to join a platoon sooner. The influence is however marginal.

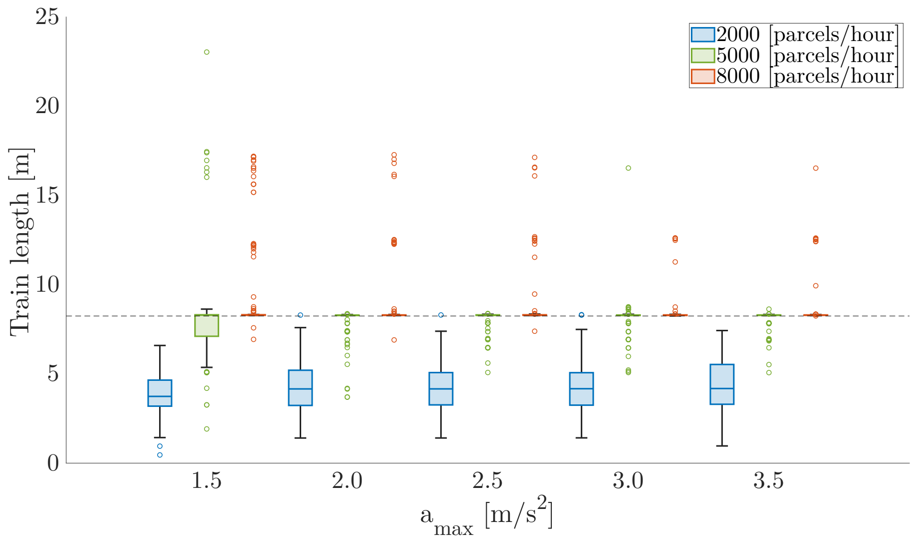


Figure 6.6. Box plot of train length for different maximum accelerations with arrival rates of 2000, 5000 and 8000 [parcels/hour].

Fourth, the maximum acceleration is considered. Here, the same holds as for the maximum velocity. A higher acceleration allows for parcels to join a platoon sooner, but the influence is again marginal.

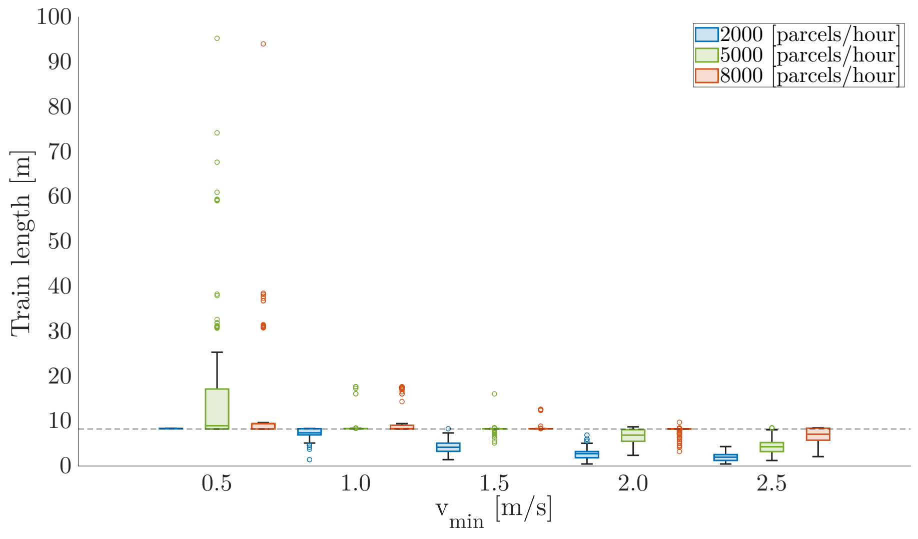


Figure 6.7. Box plot of train length for different minimum velocities with rates of 2000, 5000 and 8000 [parcels/hour].

Lastly, the minimum allowed velocity is analysed. It can be seen that the minimum velocity has a big influence on the conveyor length. This makes sense, as more time in the overflow zone means more time to join a platoon. The drawback, however, is that due to the lower velocity the density in the overflow zone increases. Due to this increase of density, excessive long trains might be formed. On itself, this is not a bad thing: the density on the merge conveyor stays high and with that the system throughput is high. However, this does prevent other infeeds to release trains to the merge zone which is undesired.

In conclusion, in the box plots, it is shown that mainly the number of conveyors and the minimum allowed velocity have an influence on the train length. This can be explained by the fact that these are the two parameters that determine the time a parcel stays in the overflow zone. The more time a parcel stays in the overflow zone, the more time there is to form a platoon of the desired length. The other three input parameters (maximum conveyor velocity, maximum conveyor acceleration and conveyor length) do not have a big influence on the time a parcel stays in the overflow zone and therefore only have little effect on the output.

When considering the three key performance indicators as mentioned in Section 2.6, the following can be concluded. Although it is desired that the footprint of the overflow zone is decreased, this measure has a direct negative influence on the platoon length. Therefore, a trade-off must be made. Regarding the power consumption, as mentioned in Section 2.6 the power consumption is mainly determined by the maximum velocity and acceleration. From the results in it can be concluded that it is not necessary to install powerful conveyors that can reach high velocities and have a high acceleration as these do not influence the performance much.

6.6 Full factorial DOE

Besides evaluating the output when altering a single design variable, it is also interesting to look at how the five different design variables interact. A possible and widely used method to do this is by means of *design of experiments* (DOE), a method of experimenting where the response of different combinations of input variables is measured. By means of design of experiment, the main effect of variables as well as the interaction between variables can be measured. The main effect describes the impact of a single variable on the output, while the interaction describes how a combination of input variables affects the output.

Design of experiments can be divided into two methods: *full factorial* design of experiments and, *fractional factorial* design of experiments (Kechagias et al., 2017). In full factorial design of experiments, all possible combinations of input variables are tested. As all combinations are tested, this gives good and reliable results. The disadvantage however is the number of experiments, or simulations in this case, that need to be run. The number of simulations to run is defined by

$$\text{Number of simulations} = I^k \tag{6.7}$$

with k being the number of input variables, often called *factors* and I being the number of values to test for each factor, often called the *levels*. So, for five variables with three values per variable, a total of $3^5 = 243$ simulations must be run. In fractional factorial design of experiments, on the other hand, not all combinations of input parameters are tested. The number of simulations to run is now defined by

$$\text{Number of simulations} = I^{k-p} \tag{6.8}$$

with p being an integer that defines the resolution of the simulations. The higher p, the fewer simulations that are run and the less accurate the results are. Fractional factorial design of experiments is thus faster, but the output result is not always reliable. In addition, aliasing might occur in fractional factorial design of experiments, resulting in inexplicable responses.

With the advantages and disadvantages of both the full and fractional factorial design of experiments stated, a decision can be made on which one to use. As the simulation time is quite short, it is chosen to perform to do a full factorial design of experiments as this yields better results and avoids aliasing. It is chosen to assign the following input parameters:

Table 6.4. Range of input parameters used for the construction of the interaction plot.

Input parameter	**Value**	**Unit**	**Type of parameter**
Arrival rate	2000, 5000, 8000	[parcels/hour]	System parameter
Number of conveyors	40, 60, 80	[-]	Design parameter
Conveyor length	0.4, 0.6, 0.8	[m]	Design parameter
Maximum conveyor velocity	2.5, 3.5, 4.5	[m/s]	Design parameter
Maximum conveyor acceleration	1.5, 2.5, 3.5	[m/s^2]	Design parameter
Minimum conveyor velocity	0.5, 1.5, 2.5	[m/s]	Control parameter

With those input parameters, the resulting average number of parcels per train can be computed. The results can thereafter be visualized in a so-called *interaction plot*. An interaction plot is a plot consisting of $k \times k$ subplots, with k still being the number of factors. Each plot shows the average number of parcels per train for different input parameters. Each row of subplots shows the output for the levels of a single variable in combination with the variables defined in each column. To interpret the interaction plot correctly, one should know what to look for. First, there is the tangent of a line. The steeper a line, the greater the influence the variable of that column has on the main effect. A horizontal line, therefore, implies that an input variable has no influence on the outcome. Second, one should look for intersections. An intersection means that there is interaction between two input variables. The more nonparallel the lines are, the greater the interaction is between them. In Figure 6.8, a simple example of an interaction plot can be seen to show how an interaction plot can be interpreted.

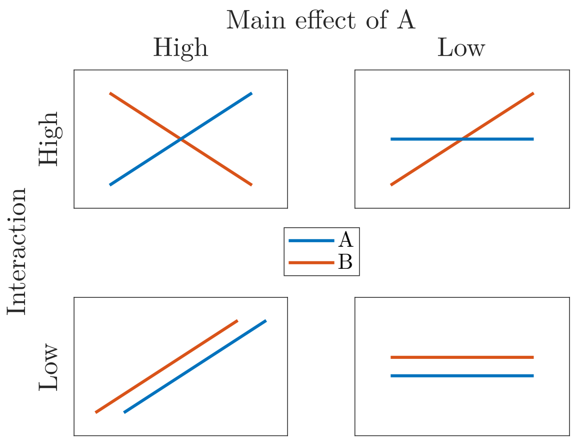


Figure 6.8. Simple example of an interaction plot.

In Figure 6.8, four interaction plots can be seen. The plots in the column on the left hand side show a high main effect of variable 'A', and the plots on the right hand side column show a low main effect of variable 'A'. This can seen as the lines of variable 'A' are steep on the left hand side while they are (almost) horizontal on the right hand side. Furthermore, in the upper two plots a high interaction between variable 'A' and 'B' can be seen, as the lines cross and are (almost) perpendicular. In the lower two plots, on the other hand, low interaction between variable 'A' and 'B' can be seen as the lines are (almost) parallel. With this in mind, the interaction plot with the input parameters as defined in Table 6.4 as shown in Figure 6.9 can be interpreted.

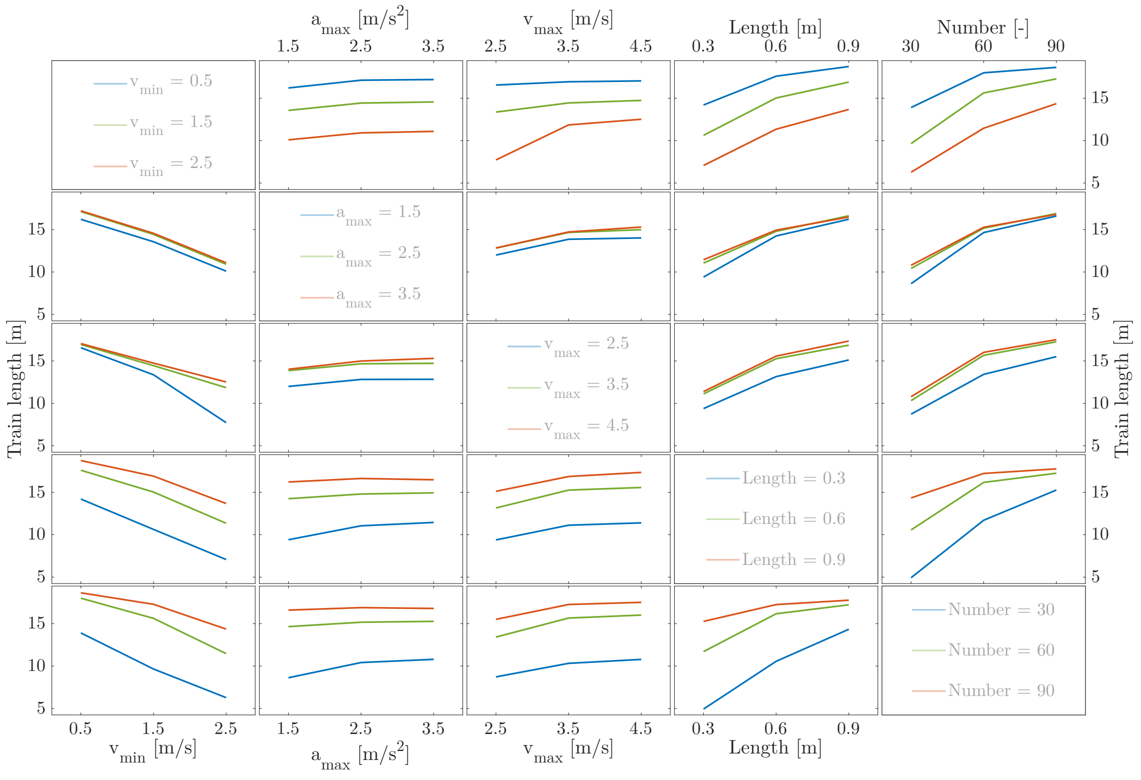


Figure 6.9. Interaction plot of five input parameters with an arrival rate of 5000 [parcels/hour].

From Figure 6.9 two conclusions can be drawn. First, the minimum velocity in combination with the overflow length have the most influence on the train length. It is clear that lines in the first, fourth and fifth column of subplots are steeper compared with the lines in the other subplots. As explained above, the steeper the lines, the greater the influence of the variable of that column on the main effect. Second, there are no intersections that are perpendicular or close to perpendicular. This implies that the input variables do not interact much regarding the average number of parcels per train.

In Appendix C, the interaction plot for an arrival rate of 2000 and 8000 [parcels/hour] can be seen. These interaction plots support the drawn conclusions.

7 Conclusion and recommendation

In this project, the research question

> *"Can a platooning algorithm be developed for real-time control of an overflow zone capable of decelerating, accumulating and releasing a train of trays or totes?"*

is answered. To do so, first, the problem is identified and the key performance indicators are formulated. Next, a literature review is performed to investigate current solutions. The method to design trajectories as described in (Timmerman and Boon, 2021) looks promising to tackle the problem at hand, therefore this method is looked into in more detail. Next, a real-time controller is developed that designs trajectories for parcels based on the method introduced by (Timmerman and Boon, 2021) and actively controls the conveyors that carry the parcels. With this controller working, the second research question

> *"What is the influence of design parameters of the overflow zone controlled by a platooning algorithm on the key performance indicators of the Posisorter?"*

can be answered. To investigate the influence of design parameters, simulations are performed with different input parameters. In the end, main effect plots, as well as interaction plots, are shown to answer the sub-question. In this chapter, the conclusion and recommendations for future research are discussed.

7.1 Conclusion

First of all, it has been shown that a real-time platooning controller for the overflow zone that fulfills all requirements can be developed in a model-based design. By designing trajectories based on acceleration and deceleration times that are calculated with closed-form expressions, computationally expensive algorithms with (optimization) solvers can be left out. This results in a fast controller, that is computationally efficient. This makes the controller fast and therefore usable in real-time situations.

Now it is concluded that the design of a real-time controller for computer simulations is feasible, the influence of design parameters on the key performance indicators of the overflow zone can be discussed. Based on simulations to find the main effect as well as the interaction between design parameters, the following can be concluded.

The ability to form a platoon of the desired length is mainly based on the time a parcel spends in the overflow zone. If the overflow zone is longer or the minimum allowed velocity is lower, there is more time for parcels that arrive to join an existing platoon and therefore the platoons formed come closer to the desired platoon length. A longer overflow zone also is able to deal better with a high arrival rate, as there is more space to collect all incoming parcels. A longer overflow zone is unfortunately not desired, therefore a trade-off must be made.

As the ability to form a platoon is mainly based on the platoon length and minimum allowed velocity, the maximum conveyor velocity and acceleration do not have a big influence. This is good news for the key performance indicator of low power consumption. As explained in Section 2.6, a lower power consumption generally also means a less expensive and less noisy conveyor. As the maximum velocity and acceleration directly affect the power consumption of a conveyor, where a higher maximum velocity and acceleration consume more energy, it is desired to have a relatively low maximum velocity and acceleration. As the effect of low maximum velocity and acceleration on forming a platoon is minimal, cheaper conveyors with a low power consumption will do just fine.

7.2 Recommendation

First of all, the assumption that the controller has full observability is made. This means, that the controller at every timestep knows the position and velocity of parcels. Although it is possible to design a full observable overflow zone in a real-life situation, this requires an expensive and comprehensive system. Currently, the overflow zone uses a number of photoelectric sensors that measure the presence of a parcel. With those measurements, in combination with the conveyor velocities, the expected parcel position is determined over time. Every time a sensor measures the presence of a parcel, the position of that parcel is updated.

Although the platooning controller does work with expected parcel positions and velocities, the performance of it is not investigated. To obtain a better insight into the performance in a real-life situation using only photoelectric sensors, future research must be done.

Second, the integration of the overflow zone and the merge zone is not considered in this project. In the current method, platoons are formed and delivered to the merge zone as late as possible. Even if a platoon is already of the desired length, it does not communicate with the merge zone on the optimal departure time. On system level it might be better to advance the departure time of a platoon such that, in combination with other infeeds, all windows on the merge conveyors are filled. By doing this, the density on the merge zone can be increased and with that the system throughput.

Last, it is recommended to test the current control model on a real system. Although the plant model has been modelled with care, the actual behaviour of parcels on conveyors is not known exactly. Testing the algorithm on a real-life system can give insights into possible shortcomings of both the plant and controller model.

Conditions for trajectory design

A

A.1 Supporting equations for ideal trajectory calculation

The three function that form the ideal trajectory as shown in Figure 5.6 over time t [s] are given in Equation A.1, Equation A.2 and Equation A.3 as

$$x_{\text{idealv}_{\text{min}}}(t) = v_{\text{min}} \cdot t + x_{\text{startacc}} - v_{\text{min}} \cdot t_{\text{startacc}} \tag{A.1}$$

$$x_{\text{idealacc}}(t) = x_{\text{startacc}} + \tfrac{1}{2} \cdot a_{\text{max}} \cdot (t - t_{\text{startacc}})^2 + v_{\text{min}} \cdot (t - t_{\text{startacc}}) \tag{A.2}$$

$$x_{\text{idealv}_{\text{final}}}(t) = v_{\text{final}} \cdot t - v_{\text{final}} \cdot t_{\text{final}} \tag{A.3}$$

The position the platoon leader starts accelerating is given by x_{startacc} [m] and defined as

$$x_{\text{startacc}} = -(t_{\text{final}} - t_{\text{startv}_{\text{final}}}) \cdot v_{\text{final}} - v_{\text{min}} \cdot t_{\text{v}_{\text{min}} \to \text{v}_{\text{final}}} - \tfrac{1}{2} \cdot a_{\text{max}} \cdot t^2_{\text{v}_{\text{min}} \to \text{v}_{\text{final}}} \tag{A.4}$$

as can also be seen in Figure 5.6.

A.2 Supporting equations for decline border calculations

The two equations used to calculate the times t_1 and t_2 [s] as shown in Figure 5.7 are

$$t_1 = \frac{2 \cdot v_{\text{min}} - v_{\text{final}} + a_{\text{max}} \cdot t_{\text{startv}_{\text{final}}} - v_{\text{max}}}{a_{\text{max}}} + \frac{2 \cdot \sqrt{(-\tfrac{3}{2} + v_{\text{min}}) - (\tfrac{1}{2} + 2 \cdot v_{\text{min}}) \cdot v_{\text{max}}}}{a_{\text{max}}} \tag{A.5}$$

$$t_2 = \frac{v_{\text{final}} + a_{\text{max}} \cdot t_{\text{startv}_{\text{final}}} - v_{\text{max}}}{a_{\text{max}}} + \frac{2 \cdot \sqrt{a_{\text{max}} \cdot x_{\text{startacc}} + a_{\text{max}} \cdot v_{\text{final}} \cdot (-t_{\text{final}} + t_{\text{startv}_{\text{final}}}) + \frac{v_{\text{max}} - v^2_{\text{final}} - v^2_{\text{max}} + v^2_{\text{min}}}{2}}}{a_{\text{max}}} \tag{A.6}$$

The functions that form x_{decline} as also shown in Figure 5.7 are

$$x_{\text{decline}}(t_{\text{reachv}_{\text{max}}}) = x_{\text{startacc}} + v_{\text{min}} \cdot (t_{\text{reachv}_{\text{max}}} - t_{\text{v}_{\text{final}}}) + \frac{2 \cdot v_{\text{min}} \cdot (v_{\text{final}} + v_{\text{max}}) - v^2_{\text{max}} - 3 \cdot v^2_{\text{min}}}{2 \cdot a_{\text{max}}} \tag{A.7}$$

$$\begin{aligned} x_{\text{decline}}(t_{\text{reachv}_{\text{max}}}) = {} & x_{\text{startacc}} - (\tfrac{1}{2} \cdot a_{\text{max}} \cdot (t_{\text{v}_{\text{max}} \to \text{v}_{\text{min}}})^2 + v_{\text{min}} \cdot t_{\text{v}_{\text{max}} \to \text{v}_{\text{min}}}) \\ & + \tfrac{1}{4} \cdot a_{\text{max}} \cdot (t_{\text{reachv}_{\text{max}}} - t_{\text{startacc}} + t_{\text{v}_{\text{max}} \to \text{v}_{\text{min}}})^2 \\ & + v_{\text{min}} \cdot (t_{\text{reachv}_{\text{max}}} - t_{\text{startacc}} + t_{\text{v}_{\text{max}} \to \text{v}_{\text{min}}}) \end{aligned} \tag{A.8}$$

$$x_{\text{decline}}(t_{\text{reachv}_{\text{max}}}) = -\frac{v^2_{\text{final}} - 2 \cdot v_{\text{final}} \cdot v_{\text{max}} + v^2_{\text{max}} + 2 \cdot a_{\text{max}} \cdot v_{\text{final}} \cdot (t_{\text{final}} - t_{\text{reachv}_{\text{max}}})}{2 \cdot a_{\text{max}}} \tag{A.9}$$

where $t_{\mathrm{v_{max}\to v_{min}}}$ is time needed to decelerate from maximum velocity to minimum velocity defined as

$$t_{\mathrm{v_{max}\to v_{min}}} = \frac{v_{\mathrm{max}} - v_{\mathrm{min}}}{a_{\mathrm{max}}} \tag{A.10}$$

A.3 Supporting equations for maximum velocity conditions

The three times t_{A}, t_{B} and t_{C} that define which v_{max} collection must be used, as can be seen in Figure 5.8 are

$$\begin{aligned} t_{\mathrm{A}} = & -\frac{2\cdot a_{\mathrm{max}}\cdot(x_{\mathrm{startacc}} - x_{\mathrm{v_{max}}} - t_{\mathrm{v_{final}}}\cdot v_{\mathrm{max}}) + v_{\mathrm{max}}\cdot(2\cdot v_{\mathrm{final}} - 4\cdot v_{\mathrm{min}}) + v_{\mathrm{max}}^2}{2\cdot a_{\mathrm{max}}\cdot v_{\mathrm{max}}} \\ & + \frac{v_{\mathrm{min}}^2 + 2\cdot\sqrt{2}\cdot(v_{\mathrm{max}} - v_{\mathrm{min}})\cdot\sqrt{2\cdot v_{\mathrm{min}} - v_{\mathrm{max}} - 4\cdot v_{\mathrm{max}}\cdot v_{\mathrm{min}} + 3\cdot v_{\mathrm{min}}^2}}{2\cdot a_{\mathrm{max}}\cdot v_{\mathrm{max}}} \end{aligned} \tag{A.11}$$

$$\begin{aligned} t_{\mathrm{B}} = & \frac{2\cdot a_{\mathrm{max}}(x_{\mathrm{vmax}} + t_{\mathrm{final}}\cdot v_{\mathrm{final}} - t_{\mathrm{v_{final}}}\cdot v_{\mathrm{final}} + t_{\mathrm{v_{final}}}\cdot v_{\mathrm{max}}) + 2\cdot v_{\mathrm{final}}\cdot v_{\mathrm{max}} - v_{\mathrm{final}}^2}{2\cdot a_{\mathrm{max}}\cdot v_{\mathrm{max}}} \\ & - \frac{v_{\mathrm{max}}^2 - 2\cdot\sqrt{2}\cdot(v_{\mathrm{final}} + v_{\mathrm{max}})\cdot\sqrt{\begin{array}{r} 2\cdot a_{\mathrm{max}}\cdot(t_{\mathrm{v_{final}}}\cdot v_{\mathrm{final}} - x_{\mathrm{startacc}} - t_{\mathrm{final}}\cdot v_{\mathrm{final}}) \\ - v_{\mathrm{final}}^2 - v_{\mathrm{max}}^2 + v_{\mathrm{min}}^2 + v_{\mathrm{max}} \end{array}}}{2\cdot a_{\mathrm{max}}\cdot v_{\mathrm{max}}} \end{aligned} \tag{A.12}$$

$$t_{\mathrm{C}} = \frac{-v_{\mathrm{final}}^2 + 2\cdot v_{\mathrm{final}}\cdot v_{\mathrm{max}} - v_{\mathrm{max}}^2 + 2\cdot a_{\mathrm{max}}\cdot t_{\mathrm{final}}\cdot v_{\mathrm{max}} + 2\cdot a_{\mathrm{max}}\cdot x_{\mathrm{v_{max}}}}{2\cdot a_{\mathrm{max}}\cdot v_{\mathrm{max}}} \tag{A.13}$$

A.4 Supporting equations for S-curved trajectories calculations

Condition for collection 4: join time of ideal minimum velocity tangent

The earliest time a parcel can join the ideal minimum velocity tangent with a s-shaped curve $t_{\mathrm{joinv_{min}}}$ [s] is given by

$$\begin{aligned} t_{\mathrm{joinv_{min}}} = & t_{\mathrm{controllable}} + \frac{v_0 - v_{\mathrm{min}}}{a_{\mathrm{max}}} + 2\cdot\frac{2\cdot v_{\mathrm{min}} - 2\cdot v_0}{2\cdot a_{\mathrm{max}}} \\ & + \frac{\sqrt{2}\cdot\sqrt{\begin{array}{r} -4\cdot v_0\cdot v_{\mathrm{min}} + v_0^2 + 3\cdot v_{\mathrm{min}}^2 \\ +2\cdot a_{\mathrm{max}}\cdot(x_{\mathrm{acc}} + |x_{\mathrm{controllable}}| + t_{\mathrm{controllable}}\cdot v_{\mathrm{min}} \\ -t_{\mathrm{startacc}}\cdot v_{\mathrm{min}} + \frac{v_0 - v_{\mathrm{min}}}{a_{\mathrm{max}}}\cdot v_{\mathrm{min}}) \end{array}}}{2\cdot a_{\mathrm{max}}} \end{aligned} \tag{A.14}$$

Condition for collection 5: join time of ideal acceleration trajectory

The velocity at which $x_{\mathrm{decline}}(t, v_{\mathrm{intersect}})$ and $x_{\mathrm{firstacc}}(t)$ intersect is given by

$$\begin{aligned} v_{\mathrm{intersect}} = & \frac{(\frac{v_0}{2} - \frac{v_{\mathrm{min}}}{2} + \frac{a_{\mathrm{max}}\cdot t_{\mathrm{controllable}}}{2} + \frac{a_{\mathrm{max}}\cdot t_{\mathrm{startacc}}}{2} - a_{\mathrm{max}}\cdot t_{\mathrm{controllable}})^2 + v_0\cdot v_{\mathrm{min}}}{v_0 - v_{\mathrm{min}} + a_{\mathrm{max}}(t_{\mathrm{startacc}} - t_{\mathrm{controllable}})} \\ & - \frac{v_0^2 - \frac{a_{\mathrm{max}}^2\cdot t_{\mathrm{startacc}}^2}{2} + \frac{a_{\mathrm{max}}^2\cdot t_{\mathrm{controllable}}^2}{2} + a_{\mathrm{max}}^2\cdot t_{\mathrm{startacc}}\cdot t_{\mathrm{controllable}}}{v_0 - v_{\mathrm{min}} + a_{\mathrm{max}}(t_{\mathrm{startacc}} - t_{\mathrm{controllable}})} \\ & + \frac{a_{\mathrm{max}}\cdot(x_{\mathrm{controllable}} - x_{\mathrm{acc}} - t_{\mathrm{startacc}}\cdot(v_0 + v_{\mathrm{min}}) + t_{\mathrm{controllable}}\cdot(v_0 - v_{\mathrm{min}}))}{v_0 - v_{\mathrm{min}} + a_{\mathrm{max}}(t_{\mathrm{startacc}} - t_{\mathrm{controllable}})} \end{aligned} \tag{A.15}$$

$$t_{\mathrm{x_{firstacc}(t)\cap x_{decline}(t,v_{intersect})}} = \frac{v_{\mathrm{intersect}} - 2\cdot v_0 + v_{\min} - a_{\max}\cdot t_{\mathrm{startacc}} + 2\cdot a_{\max}\cdot t_{\mathrm{controllable}}}{a_{\max}} + \frac{2\cdot\sqrt{\begin{aligned} &-v_0\cdot v_{\mathrm{intersect}} - v_0\cdot v_{\min} + v_{\mathrm{intersect}}\cdot v_{\min} + v_0^2 \\ &+a_{\max}\cdot(x_{\mathrm{acc}} - x_{\mathrm{controllable}} + t_{\mathrm{startacc}}(v_0 - v_{\mathrm{intersect}} - v_{\min}) \\ &-t_{\mathrm{controllable}}(v_0 - v_{\mathrm{intersect}} - v_{\min})) \\ &+\tfrac{a_{\max}^2\cdot t_{\mathrm{startacc}}^2}{2} + \tfrac{a_{\max}^2\cdot t_{\mathrm{controllable}}^2}{2} - a_{\max}^2\cdot t_{\mathrm{startacc}}\cdot t_{\mathrm{controllable}}\end{aligned}}}{a_{\max}} \tag{A.16}$$

$$t_{\mathrm{joinacc}} = \frac{v_{\mathrm{intersect}} - v_{\min} + a_{\max}\cdot t_{\mathrm{startacc}} + a_{\max}\cdot t_{\mathrm{dec}}}{2\cdot a_{\max}} + \frac{\sqrt{\begin{aligned} &-2\cdot v_{\mathrm{intersect}}\cdot v_{\min} + v_{\mathrm{intersect}}^2 + v_{\min}^2 \\ &+2\cdot a_{\max}\cdot(2\cdot x_{\mathrm{x_{firstacc}(t)\cap x_{decline}(t,v_{intersect})}} - 2\cdot x_{\mathrm{acc}} + t_{\mathrm{startacc}}\cdot v_{\mathrm{intersect}} + t_{\mathrm{startacc}}\cdot v_{\min} \\ &-t_{\mathrm{dec}}\cdot v_{\mathrm{intersect}} - t_{\mathrm{dec}}\cdot v_{\min}) - a_{\max}^2(2\cdot t_{\mathrm{startacc}}\cdot t_{\mathrm{dec}} - t_{\mathrm{startacc}}^2 - t_{\mathrm{dec}}^2)\end{aligned}}}{2\cdot a_{\max}} \tag{A.17}$$

with $x_{\mathrm{x_{firstacc}(t)\cap x_{decline}(t,v_{intersect})}}$ [m] equal to

$$x_{\mathrm{x_{firstacc}(t)\cap x_{decline}(t,v_{intersect})}} = x_{\mathrm{controllable}} + v_0\cdot t_{\mathrm{v_{intersect}\to v_{intersect}}} + \frac{1}{2}\cdot a_{\max}\cdot\left(t_{\mathrm{v_{intersect}\to v_{intersect}}}\right)^2 \tag{A.18}$$

with

$$t_{\mathrm{v_{intersect}\to v_{intersect}}} = \frac{v_{\mathrm{intersect}} - v_0}{a_{\max}} \tag{A.19}$$

Condition for collection 6: join time of ideal final velocity tangent

The time when x_{control} and $x_{\mathrm{v_{final}}}$ intersect is given by

$$t_{\mathrm{x_{control}\cap x_{v_{final}}}} = \frac{x_{\mathrm{control}}}{v_{\mathrm{final}}} + t_{\mathrm{final}} \tag{A.20}$$

and is used to calculate $t_{\mathrm{joinv_{final}}}$ which is determined by

$$t_{\mathrm{joinv_{final}}} = t_{\mathrm{controllable}} + (v_{\mathrm{final}} - v_0)/a_{\max} + 2\cdot\frac{\sqrt{2}\cdot\sqrt{\begin{aligned} &-(v_0 - v_{\mathrm{final}})^2 - 2\cdot v_{\mathrm{final}}\cdot|(v_0 - v_{\mathrm{final}})| \\ &+2\cdot a_{\max}\cdot(x_{\mathrm{control}} - x_{\mathrm{controllable}} + t_0\cdot v_{\mathrm{final}} - t_{\mathrm{crossxctrlvfinal}}\cdot v_{\mathrm{final}} \\ &+|t_{\mathrm{v_0\to v_{final}}}|\cdot v_{\mathrm{final}})\end{aligned}}}{2\cdot a_{\max}} \tag{A.21}$$

Supporting calculations for collections for trajectory design

B

B.1 Collection 1

First, the tangent of the maximum velocity trajectory of a parcel $x_{\mathrm{v}_{\max}}(t)$ [m]is given by

$$x_{\mathrm{v}_{\max}}(t) = v_{\max} \cdot t + x_{\mathrm{controllable}} + \tfrac{1}{2} \cdot a_{\max} \cdot t^2_{\mathrm{v}_0 \to \mathrm{v}_{\max}} + v_0 \cdot t_{\mathrm{v}_0 \to \mathrm{v}_{\max}} - (t_{\mathrm{controllable}} + t_{\mathrm{v}_0 \to \mathrm{v}_{\max}}) \cdot v_{\max} \tag{B.1}$$

The time at which x_{vmax} and x_{ideal} cross in collection 1 as shown in Figure 5.9 is given by

$$t_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{v}_{\min}}} = -(x_{\mathrm{controllable}} - x_{\mathrm{startacc}} + \frac{(v_0 - v_{\max})^2}{2 \cdot a_{\max}} - v_{\max} \cdot (t_{\mathrm{controllable}} - t_{\mathrm{v}_0 \to \mathrm{v}_{\max}}) + v_{\min} \cdot (t_{\mathrm{startv}_{\mathrm{final}}} - t_{\mathrm{v}_{\min} \to \mathrm{v}_{\mathrm{final}}}) - \frac{v_0 \cdot t_{\mathrm{v}_0 \to \mathrm{v}_{\max}}}{v_{\max} - v_{\min}} \tag{B.2}$$

where $t_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{v}_{\min}}}$ is in [m]. The time needed to decelerate from maximum velocity to minimum velocity $t_{\mathrm{v}_{\max} \to \mathrm{v}_{\min}}$ as also shown in Figure 5.9 is given by Equation A.10.

B.2 Collection 2

The time $t_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{decline}}}$ represents the time when $x_{\mathrm{v}_{\max}}(t)$ and $x_{\mathrm{decline}}(t)$ intersect. This is given by

$$t_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{decline}}} = \frac{2 \cdot v_{\max} - 2 \cdot v_{\min} + a_{\max} \cdot (t_{\mathrm{startacc}} - t_{\mathrm{vmax} \to \mathrm{v}_{\min}})}{a_{\max}} - \frac{2 \cdot \sqrt{\begin{array}{r} a_{\max} \cdot (x_{\mathrm{controllable}} - x_{\mathrm{startacc}}) \\ a_{\max} \cdot v_{\max} \cdot (t_{\mathrm{startacc}} - t_{\mathrm{controllable}} - t_{\mathrm{vmax} \to \mathrm{v}_{\min}} - t_{\mathrm{v0} \to \mathrm{vmax}}) \\ + a_{\max} \cdot (\frac{1}{2} \cdot a_{\max} \cdot t^2_{\mathrm{v0} \to \mathrm{vmax}} + v_0 \cdot t_{\mathrm{v0} \to \mathrm{vmax}}) \\ + a_{\max} \cdot (-2 \cdot v_{\max} \cdot v_{\min} + \frac{3}{2} \cdot v^2_{\max} + \frac{1}{2} v^2_{\min}) \end{array}}}{a_{\max}} \tag{B.3}$$

The deceleration trajectory from $t_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{decline}}}$ to the ideal acceleration trajectory x_{ideal} is given by

$$x_{\mathrm{dec2ideal}} = x_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{decline}}} - \tfrac{1}{2} \cdot (a_{\max} \cdot (t_{\mathrm{v}_{\min}} - t_{\mathrm{dec}})^2 + v_{\max} \cdot (t_{\mathrm{v}_{\min}} - t_{\mathrm{dec}}) \tag{B.4}$$

Note that $t_{\mathrm{v}_{\min}}$ is still unknown at this point and $x_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{decline}}}$ is given by

$$x_{\mathrm{x}_{\mathrm{v}_{\max}} \cap \mathrm{x}_{\mathrm{decline}}} = x_{\mathrm{startacc}} - (\tfrac{1}{2} \cdot a_{\max} \cdot t^2_{\mathrm{vmax} \to \mathrm{v}_{\min}} + v_{\min} \cdot t_{\mathrm{vmax} \to \mathrm{v}_{\min}}) + \tfrac{1}{4} \cdot a_{\max} \cdot (t_{\mathrm{dec}} - (t_{\mathrm{startacc}} - t_{\mathrm{vmax} \to \mathrm{v}_{\min}}))^2 + v_{\min} \cdot (t_{\mathrm{dec}} - (t_{\mathrm{startacc}} - t_{\mathrm{vmax} \to \mathrm{v}_{\min}})) \tag{B.5}$$

With the use of $x_{\text{dec2ideal}}$, the time on which the decelerating trajectory intersects with the ideal accelerating trajectory is given by

$$
\begin{aligned}
t_{\mathrm{x_{dec2ideal}} \cap \mathrm{x_{ideal}}} = & \frac{v_{\max} - v_{\min} + a_{\max} \cdot t_{\text{startacc}} + a_{\max} \cdot t_{\text{dec}}}{2 \cdot a_{\max}} \\
& + \frac{\sqrt{\begin{aligned} -2 \cdot v_{\max} \cdot v_{\min} + v_{\max}^2 + v_{\min}^2 \\ + a_{\max}^2 (-t_{\text{startacc}}^2 - t_{\text{dec}}^2 - 2 \cdot t_{\text{startacc}} \cdot t_{\text{dec}}) \\ +2 \cdot a_{\max} (t_{\text{startacc}} \cdot (v_{\max} + v_{\min}) - t_{\text{dec}} \cdot (v_{\max} + v_{\min})) \\ +4 \cdot a_{\max} (-x_{\text{startacc}} + t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{decline}}}) \end{aligned}}}{2 \cdot a_{\max}}
\end{aligned}
\tag{B.6}
$$

B.3 Collection 3

The intersection between xvmax as given in collection 1 and xideal as given in condition 1 is given by

$$
\begin{aligned}
t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{v_{final}}}} = & - \frac{(t_{\text{final}} - t_{\text{startvfinal}}) \cdot v_{\text{final}}}{v_{\max} - vf} \\
& - \frac{(x_{\text{controllable}} + \frac{1}{2} \cdot a_{\max} \cdot t_{\max \to \mathrm{v0}}^2 + v0 \cdot t_{\mathrm{v_{max}} \to \mathrm{v0}})}{v_{\max} - vf} \\
& + \frac{t_{\text{ctrl}} + t_{\text{vmax} \to \text{v0}} \cdot v_{\max} - t_{\text{startvfinal}} \cdot vf}{v_{\max} - vf}
\end{aligned}
\tag{B.7}
$$

where $t_{\mathrm{x_{v_{max}}} \cap \mathrm{x_{v_{final}}}}$ is in [m]. The time needed to decelerate from maximum velocity to final velocity is given by

$$
t_{\mathrm{v_{vmax}} \to \mathrm{v_{final}}} = \frac{v_{\max} - v_{\text{final}}}{a_{\max}} \tag{B.8}
$$

where $t_{\mathrm{v_{vmax}} \to \mathrm{v_{final}}}$ is in [s].

B.4 Collection 4

The distance s_{scurve} the s-curved shape must cover is defined by

$$
\begin{aligned}
s_{\text{scurve}} = & |x_{\text{controllable}}| - v_{\min} \cdot (t_{\text{startacc}} - t_{\text{controllable}} - t_{\mathrm{v_0} \to \mathrm{v_{min}}} - 2 \cdot t_{\text{scurve}}) \\
& + x_{\text{startacc}} - (\tfrac{1}{2} \cdot a_{\max} \cdot t_{\mathrm{v0} \to \mathrm{v_{min}}}^2 + v_{\min} \cdot t_{\mathrm{v0} \to \mathrm{v_{min}}})
\end{aligned}
\tag{B.9}
$$

and the time the s-curved shape takes is defined by

$$
\begin{aligned}
tscurve = & \frac{2 \cdot v_{\min} - 2 \cdot v_0}{2 \cdot a_{\max}} \\
& + \frac{\sqrt{2} \cdot \sqrt{\begin{aligned} 2 \cdot a_{\max} (x_{\text{startacc}} + |x_{\text{controllable}}| + v_{\min} (t_{\text{controllable}} + t_{\mathrm{v0} \to \mathrm{v_{min}}} - t_{\text{startacc}})) \\ + v_0^2 + 3 \cdot v_{\min}^2 - 4 \cdot v_0 \cdot v_{\min} \end{aligned}}}{2 \cdot a_{\max}}
\end{aligned}
\tag{B.10}
$$

B.5 Collection 5

The equation that defines the position of the deceleration line is given by

$$\begin{aligned} x_{\text{decline}}(t, v_{\text{intersect}}) = x_{\text{startacc}} &- (\tfrac{1}{2} \cdot a_{\max} \cdot (t_{\text{v}_{\text{intersect}} \to \text{v}_{\min}})^2 + v_{\min} \cdot t_{\text{v}_{\text{intersect}} \to \text{v}_{\min}}) \\ &+ \tfrac{1}{4} \cdot a_{\max} \cdot (t - t_{\text{startacc}} + t_{\text{v}_{\text{intersect}} \to \text{v}_{\min}})^2 \\ &+ v_{\min} \cdot (t - t_{\text{startacc}} + t_{\text{v}_{\text{intersect}} \to \text{v}_{\min}}) \end{aligned} \tag{B.11}$$

The first acceleration trajectory of the parcel when entering the control region is defined as

$$x_{\text{firstacc}}(t) = x_{\text{controllable}} + \tfrac{1}{2} \cdot a_{\max} \cdot (t - t_{\text{controllable}})^2 + v_0 \cdot (t - t_{\text{controllable}}) \tag{B.12}$$

The equation that describes the time of intersection based on $v_{\text{intersect}}$ is defined as

$$\begin{aligned} t_{\text{intersect}}(v_{\text{intersect}}) = &\frac{v_{\text{intersect}} - 2 \cdot v0 + v_{\min} - a_{\max} \cdot t_{\text{startacc}} + 2 \cdot a_{\max} \cdot t_{\text{controllable}}}{a_{\max}} \\ &+ \frac{2 \cdot \sqrt{\begin{aligned} &a_{\max} \cdot (x_{\text{startacc}} - x_0 + t_{\text{startacc}} \cdot (v_0 - v_{\min} - v_{\text{intersect}})) \\ &+a_{\max} \cdot (t_{\text{controllable}} \cdot (v_{\min} + v_{\text{intersect}} - v_0)) \\ &+v_0 \cdot (v_0 - v_{\text{intersect}} - v_{\min}) + v_{\text{intersect}} \cdot v_{\min} \\ &+\tfrac{a_{\max}^2 \cdot t_{\text{startacc}}^2}{2} + \tfrac{a_{\max}^2 \cdot t_{\text{controllable}}^2}{2} - a_{\max}^2 \cdot t_{\text{startacc}} \cdot t_{\text{controllable}} \end{aligned}}}{a_{\max}} \end{aligned} \tag{B.13}$$

B.6 Collection 6

The time t_{scurve} the s-curved shape must cover is defined by

$$\begin{aligned} t_{\text{scurve}} = &\frac{2 \cdot v_0 - 2 \cdot v_{\text{final}}}{2 \cdot a_{\max}} \\ &+ \frac{\sqrt{2} \cdot \sqrt{\begin{aligned} &2 \cdot a_{\max}(x_{\text{controllable}} - x_{\text{control}} - t_{\text{controllable}} \cdot v_{\text{final}} \\ &+t_{\text{x}_{\text{control}} \cap \text{x}_{\text{vfinal}}} \cdot v_{\text{final}} - |t_{\text{v}_0 \to \text{vfinal}}| \cdot \max[v_0, v_0]) \\ &-4 \cdot v0 \cdot v_{\text{final}} + |v_0 - v_{\text{final}}|^2 + 2 \cdot v_0^2 + 2 \cdot v_{\text{final}}^2 + 2 \cdot \min[v_0, v_{\text{final}}] \cdot |v_0 - v_{\text{final}}| \end{aligned}}}{2 \cdot a_{\max}} \end{aligned} \tag{B.14}$$

with the distance the s-curved covers defined by

$$\begin{aligned} s_{\text{scurve}} = &v_{\text{final}} \cdot (t_{\text{v}_0 \to \text{v}_{\text{final}}} + 2 \cdot t_{\text{scurve}}) + (v_{\text{final}} \cdot (t_{\text{control}} - t_{\text{x}_{\text{control}} \cap \text{x}_{\text{vfinal}}}) + x_{\text{control}} - x_{\text{controllable}}) \\ &- (\tfrac{1}{2} \cdot a_{\max} \cdot t_{\text{v}_0 \to \text{v}_{\text{final}}}^2 + v_{\text{final}} \cdot t_{\text{v}_0 \to \text{v}_{\text{final}}}) \end{aligned} \tag{B.15}$$

Interaction plots

C

Interaction plot with input parameters as defined in Table 6.4 with arrival rates of 2000 and 8000 [parcels/hour].

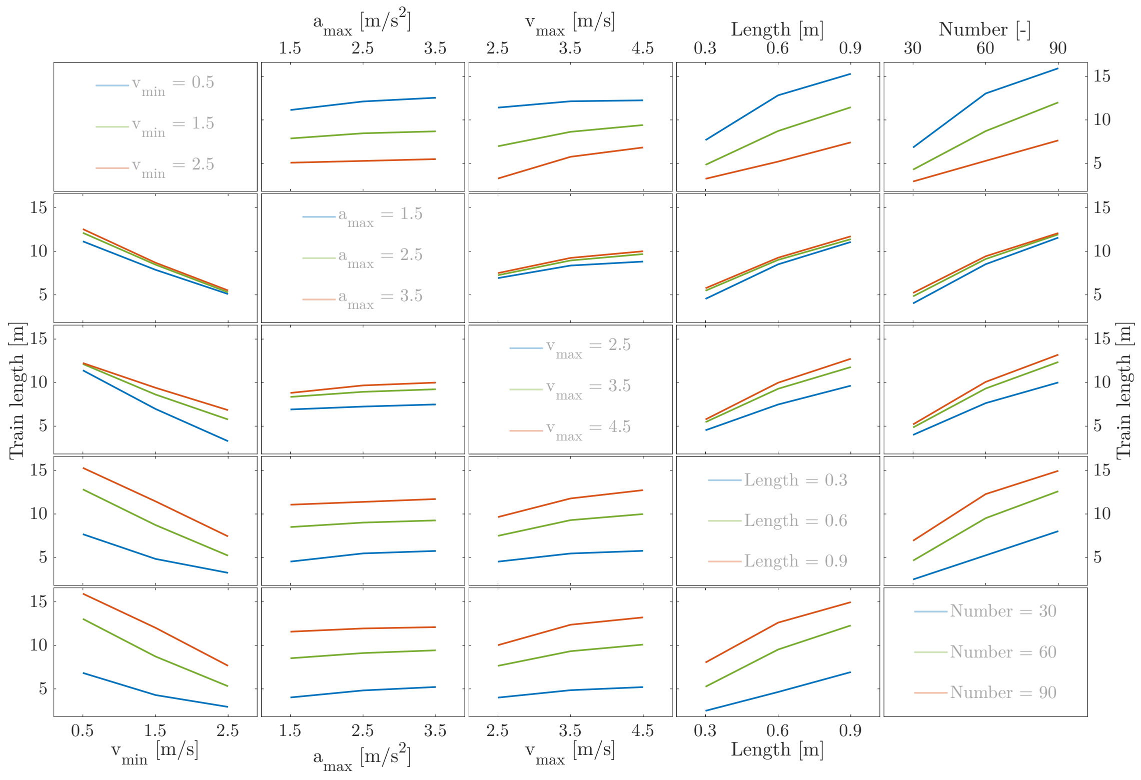


Figure C.1. Interaction plot of five input parameters with an arrival rate of 2000 [parcels/hour].

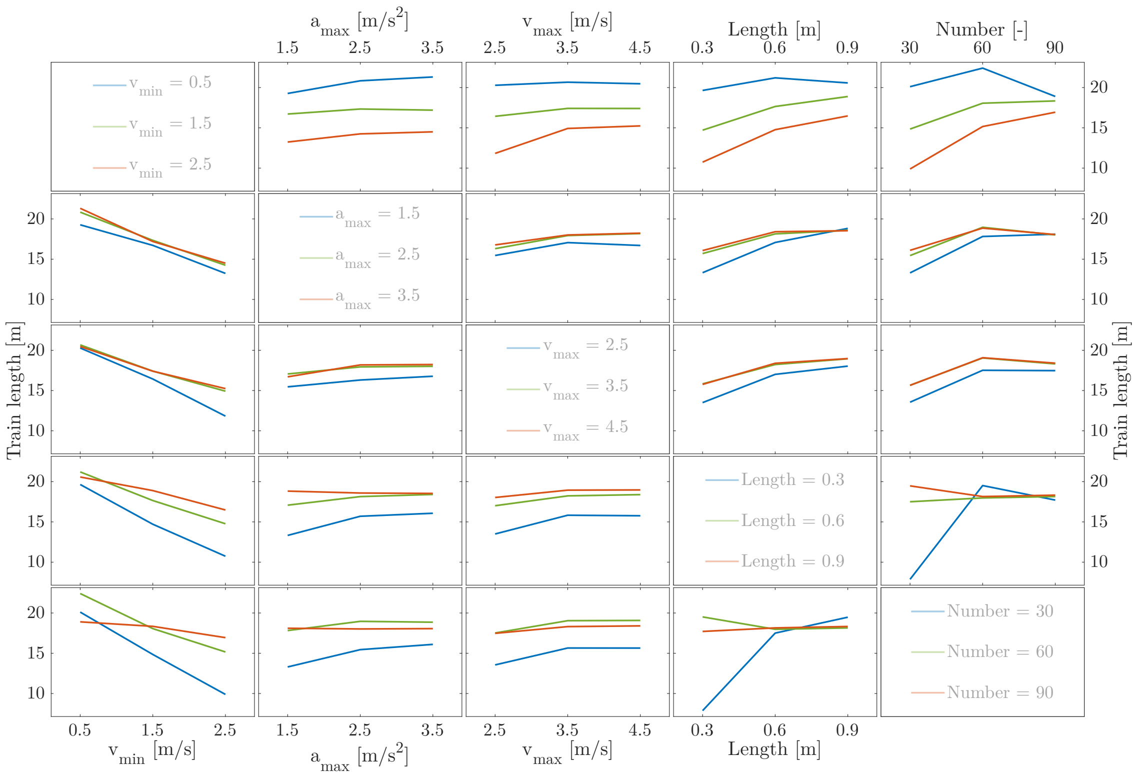


Figure C.2. Interaction plot of five input parameters with an arrival rate of 8000 [parcels/hour].

Bibliography

[1] R. Aarenstrup. *Managing Model-Based Design*. The MathWorks, Inc., 2015. ISBN 9781512036138. doi: 10.1001/archderm.1982.01650190012009.

[2] M. Ahmadian. Model based design and SDR. Technical report, Sundance Multiprocessing Technology Ltd, School of Engineering and Electronics. University Of Edinburgh, Sundance Digital Signal Processing Inc, The MathWorks, 2006.

[3] M. Alirezaei, T. J. Van Den Boom, and R. Babuska. Max-plus algebra for optimal scheduling of multiple sheets in a printer. *Proceedings of the American Control Conference*, pages 1973–1978, 2012. ISSN 07431619. doi: 10.1109/acc.2012.6315457.

[4] C. Bergenhem, H. Pettersson, E. Coelingh, C. Englund, S. Shladover, and S. Tsugawa. Overview of platooning systems. *19th Intelligent Transport Systems World Congress, ITS 2012*, 2012.

[5] A. Bergmann. Benefits and Drawbacks of Model-based Design. *KMUTNB International Journal of Applied Science and Technology*, 7(3):15–19, 2014. ISSN 1906151X. doi: 10.14416/j.ijast.2014.04.004.

[6] M. A. Boon, R. D. van der Mei, and E. M. Winands. Applications of polling systems. *Surveys in Operations Research and Management Science*, 16(2):67–82, 2011. ISSN 18767354. doi: 10.1016/j.sorms.2011.01.001.

[7] K. Dresner and P. Stone. A Multiagent Approach to Autonomous Intersection Management. *Journal of Artificial Intelligence Research*, 31(1):591–656, 2008. ISSN 02786648. doi: 10.1109/45.565609.

[8] T. E. Kalbitz. A Comparison of Approaches for Platooning Management. Technical report, Universität Mannheim, 2017.

[9] J. Kechagias, K. Kitsakis, and N. Vaxevanidis. Comparison of full versus fractional factorial experimental design for the prediction of cutting forces in turning of a titanium alloy : a case study. *International Journal of Materials*, 4(May):1–4, 2017.

[10] J. Lee and B. Park. Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):81–90, 2012. ISSN 15249050. doi: 10.1109/TITS.2011.2178836.

[11] S. Liaskos, S. A. McIlraith, S. Sohrabi, and J. Mylopoulos. Integrating preferences into goal models for requirements engineering. *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference, RE2010*, pages 135–144, 2010. doi: 10.1109/RE.2010.26.

[12] P. M. McGuire. *Conveyors: Application, Selection, and Integration*. Taylor and Francis Group, LLC, 2009. ISBN 1439803900. URL `http://books.google.com/books?id=558l93ERNU4C&pgis=1`.

[13] D. Miculescu and S. Karaman. Polling-systems-based control of high-performance provably-safe autonomous intersections. *Proceedings of the IEEE Conference on Decision and Control*, 53:1417–1423, 2014. ISSN 07431546. doi: 10.1109/CDC.2014.7039600.

[14] D. Miculescu and S. Karaman. *High Performance and Provably Safe Polling-Systems-Based Control Algorithms of All-Autonomous Intersections*. PhD thesis, Massachusetts Institute of Technology, 2015.

[15] D. Miculescu and S. Karaman. Polling-systems-based autonomous vehicle coordination in traffic intersections with no traffic signals. *IEEE Transactions on Automatic Control*, 65(2): 680–694, 2020. ISSN 15582523. doi: 10.1109/TAC.2019.2921659.

[16] R. Parasuraman and M. Mouloua. *Automation and Human Performance: Theory and Applications*. Lawrence Erlbaum Associates, Inc., 1996.

[17] J. Reedy, S. Lunzman, and B. Mekari. Model based design accelerates the development of mechanical locomotive controls. *SAE Technical Papers*, 2010. ISSN 26883627. doi: 10.4271/2010-01-1999.

[18] P. Samarati, I. Ray, and I. Ray. *From Database to Cyber Security*. Springer Nature Switzerland AG 2018, 2018. ISBN 9783030048334. doi: 10.1007/978-3-030-04834-1.

[19] M. Struik. Design Optimisation of a Throw-Catch Infeed Using a Model-Based Design Approach, MSc thesis. *Eindhoven University of Technology, Netherlands*, 2018.

[20] L. Swartjes. Model-based constrained optimization for paper path layout and timing design of printers, MSc thesis. *Eindhoven University of Technology, Netherlands*, 2012.

[21] R. Tachet, P. Santi, S. Sobolevsky, L. I. Reyes-Castro, E. Frazzoli, D. Helbing, and C. Ratti. Revisiting street intersections using slot-based systems - Supplementary Information. *PLoS ONE*, 11(3):1–13, 2016. ISSN 19326203. doi: 10.1371/journal.pone.0149607.

[22] R. Tachet, P. Santi, S. Sobolevsky, L. I. Reyes-Castro, E. Frazzoli, D. Helbing, and C. Ratti. Revisiting street intersections using slot-based systems. *PLoS ONE*, 11(3):1–9, 2016. ISSN 19326203. doi: 10.1371/journal.pone.0149607.

[23] R. W. Timmerman and M. A. Boon. Platoon forming algorithms for intelligent street intersections. *Transportmetrica A: Transport Science*, 17(3):278–307, 2021. ISSN 23249943. doi: 10.1080/23249935.2019.1692962. URL `https://doi.org/23249935.2019.1692962`.

[24] Vanderlande.com. Company profile, 2021. URL `https://www.vanderlande.com/about-vanderlande/company-profile/`.

[25] Vanderlande.com. FLEET Logistic Solution, 2021. URL `https://www.vanderlande.com/evolutions/fleet/`.

[26] Vanderlande.com. Facts and figures, 2021. URL `https://www.vanderlande.com/about-vanderlande/facts-and-figures/`.

[27] Vanderlande.com. POSISORTER, 2021. URL `https://www.vanderlande.com/systems/sortation/posisorter/`.

[28] Vanderlande.com. Sortation, 2021. URL `https://www.vanderlande.com/systems/sortation/`.

List of Figures

1.1 Vanderlandes Posisorter system (Vanderlande.com, 2021e). 3

2.1 Overview of the Posisorter system. 6
2.2 Dimensions of the side view of trays and totes. 7

3.1 Main idea of closed-form trajectory generation by Timmerman and Boon (2021). . . 18

4.1 Example plant layout with twelve short conveyors. 24
4.2 Visualization to show the behaviour of parcels on conveyors. 25
4.3 Visualisation of parcel collision behaviour. 26

5.1 Initial trajectory of platoon leader from x_{control} to 0 starting at t_0. 30
5.2 Example of collisions between parcels if two parcels arrive in a train and the departure time is not reevaluated. 30
5.3 General shape of proposed trajectory with labeled parts. 32
5.4 Visualization to show why joining the platoon as soon as possible avoids collisions. . 33
5.5 Velocity profile of trajectory similar to the trajectory shown in Figure 5.4b 34
5.6 Visualization of equations that set up the positions in time to join the platoon. 36
5.7 Visualization of times that define the equation that is used to span x_{decline}. 38
5.8 Visualization to help with determining which collection that reaches maximum velocity to choose. 38
5.9 Visualization to help with the calculations of collection 1. 39
5.10 Visualization to help with the calculations of collection 2. 40
5.11 Visualization to help with the calculations of collection 4. 42
5.12 Visualization to help with the calculations of collection 5. 43
5.13 Flow chart to decide if a parcel joins a platoon or starts its own. 46
5.14 Trajectories of parcels formed into platoons over time. 46

6.1 Average train length over time for 25 simulations with different arrival rates with mean and 2σ shown. 53
6.2 Computation time of controller for arrival rate of 2000, 5000 and 8000 [parcels/hour]. 56
6.3 Box plot of train length for different number of conveyors with arrival rates of 2000, 5000 and 8000 [parcels/hour]. 57
6.4 Box plot of train length for different conveyor lengths with arrival rates of 2000, 5000 and 8000 [parcels/hour]. 57
6.5 Box plot of train length for different maximum velocities with arrival rates of 2000, 5000 and 8000 [parcels/hour]. 58
6.6 Box plot of train length for different maximum accelerations with arrival rates of 2000, 5000 and 8000 [parcels/hour]. 58
6.7 Box plot of train length for different minimum velocities with rates of 2000, 5000 and 8000 [parcels/hour]. 59

6.8 Simple example of an interaction plot. 61
6.9 Interaction plot of five input parameters with an arrival rate of 5000 [parcels/hour]. 61

C.1 Interaction plot of five input parameters with an arrival rate of 2000 [parcels/hour]. 73
C.2 Interaction plot of five input parameters with an arrival rate of 8000 [parcels/hour]. 74

List of Tables

5.1 Decision table for closed-form expression collection. 37

6.1 Initial input parameters with chosen values for simulation. 50
6.2 Standard deviation of 25 simulations for different arrival rates. 54
6.3 Range of input parameters of which the influence on the KPIs is tested. 56
6.4 Range of input parameters used for the construction of the interaction plot. 60