Présentée par

Alessandro BOZZI

Thèse dirigée par
Roberto SACILE et
Jose-Fernando JIMENEZ et
et
Camilo HERNANDEZ-RODRIGUEZ et
et
Enrico ZERO
préparée au sein du
Laboratoire SYMME (Annecy, France) et
DELab (Genova, Italy)
Dans
l'École Doctorale Sciences, Ingénierie, Environnement et
Doctoral School in Computer Science and Systems Engineering

Platooning-based control techniques in transportation and logistic

Università degli Studi di Genova
Dipartimento di Informatica, Bioingegneria,
Robotica ed Ingegneria dei Sistemi

Platooning-based control techniques in transportation and logistic

by

Alessandro Bozzi

Theses Series **DIBRIS-TH-2023-XXXVI**

DIBRIS, Università di Genova
Via Opera Pia, 13 16145 Genova, Italy http://www.dibris.unige.it/

Università degli Studi di Genova

Dipartimento di Informatica, Bioingegneria,

Robotica ed Ingegneria dei Sistemi

Ph.D. Thesis in Computer Science and Systems Engineering
Systems Engineering Curriculum

Platooning-based control techniques in transportation and logistic

by

Alessandro Bozzi

October, 2023

Dipartimento di Informatica, Bioingegneria, Robotica ed Ingegneria dei Sistemi
Universitá degli Studi di Genova

DIBRIS, Univ. di Genova
Via Opera Pia, 13
I-16145 Genova, Italy
`http://www.dibris.unige.it/`

Ph.D. Thesis in Computer Science and Systems Engineering
Systems Engineering Curriculum
(S.S.D. ING INF/04)

Submitted by Alessandro Bozzi
Date of submission: October 2023

Title: Platooning-based control techniques in transportation and logistic

Acknowledgment

I would like to express my deepest gratitude to all the individuals who have contributed to the completion of this thesis. Without their support and encouragement, this achievement would not have been possible.

First and foremost, I wish to thank my advisor, Professor Roberto Sacile, for not only guiding me on this journey but also for granting me the privilege of collaborating on numerous projects. Your mentorship has been exceptional and you taught me much more than academic notions.

A huge thank you also goes to my advisors at the Université Savoie Mont-Blanc, Professors Jose-Fernando Jimenez and Camilo Hernandez-Rodriguez. You made this double doctoral degree possible and expanded my research horizons with your expertise.

I also want to express my gratitude to my closest colleagues, Prof. Enrico Zero and Dr. Simone Graffione. Over these three years, you have been invaluable colleagues and someone who made the lab a more enjoyable place day after day.

More broadly, I want to thank all the professors I have worked with and shared positive moments with, such as Professor Simona Sacone, Professor Silvia Siri, Professor Cecilia Pasquale, and Professor Christine Galez. Thank you for being part of my journey and teaching me inestimable insights.

The same goes for all the fellow phd and master students I have shared the lab with during these years. I would like to personally thank you for making my everyday life at Delab and Symme more agreeable.

I also want to thank all the people who have enriched my extra-academic life. A huge thank you to my flatmate Pietro for all the moments shared in our home and for being a wonderful friend. The same goes for Mari, Julian, Andreina, and Paglia. You have given me unforgettable moments over these three years, giving me happiness and gratefulness just because I have you by my side. You are like a second family to me.

Thanks to Chiara for bringing serenity into my life. You have a kind heart, and your purity of spirit helps me become a better person day by day.

Thanks to all my friends in Genoa for the moments we have spent together, and because I know I can always count on you. I wish I could thank each of you individually for everything you do for me. In the same way, thanks to my friends in Annecy. You managed to enter my life in a very short time and make my stay there fantastic. I couldn't have asked for anything better.

Finally, heartfelt thanks to my mother and my father, who have always supported me and made me the man I am today. Thank you, I love you. A final thank you goes to my brother Lorenzo. Simply, thank you for being the most important person in my life.

Abstract

This thesis explores the integration of autonomous vehicle technology with smart manufacturing systems. At first, essential control methods for autonomous vehicles, including Linear Matrix Inequalities (LMIs), Linear Quadratic Regulation (LQR)/Linear Quadratic Tracking (LQT), PID controllers, and dynamic control logic via flowcharts, are examined. These techniques are adapted for platooning to enhance coordination, safety, and efficiency within vehicle fleets, and various scenarios are analyzed to confirm their effectiveness in achieving predetermined performance goals such as inter-vehicle distance and fuel consumption. A first approach on simplified hardware, yet realistic to model the vehicle's behavior, is treated to further prove the theoretical results.
Subsequently, performance improvement in smart manufacturing systems (SMS) is treated. The focus is placed on offline and online scheduling techniques exploiting Mixed Integer Linear Programming (MILP) to model the shop floor and Model Predictive Control (MPC) to adapt scheduling to unforeseen events, in order to understand how optimization algorithms and decision-making frameworks can transform resource allocation and production processes, ultimately improving manufacturing efficiency.
In the final part of the work, platooning techniques are employed within SMS. Autonomous Guided Vehicles (AGVs) are reimagined as autonomous vehicles, grouping them within platoon formations according to different criteria, and controlled to avoid collisions while carrying out production orders. This strategic integration applies platooning principles to transform AGV logistics within the SMS. The impact of AGV platooning on key performance metrics, such as makespan, is devised, providing insights into optimizing manufacturing processes.

Throughout this work, various research fields are examined, with intersecting future technologies from precise control in autonomous vehicles to the coordination of manufacturing resources. This thesis provides a comprehensive view of how optimization and automation can reshape efficiency and productivity not only in the domain of autonomous vehicles but also in manufacturing.

Table of Contents

Chapter 1

Introduction and state of the art

In the context of contemporary technological advancements, the intersection of autonomous systems and manufacturing processes becomes a focal point of exploration. This thesis sets out to investigate the enhancement of platooning performance, the optimization of manufacturing scheduling, and the comprehensive analysis of the effects induced by platooning-based control techniques in manufacturing.
Firstly, autonomous vehicles are considered, and specifically their coordination within platoons. The objective is to identify potential improvements in platooning performance, with a particular focus on aspects like fuel efficiency and safety through platooning control.
Secondly, the allocation of tasks in manufacturing systems is treated, where the primary focus centers on refining product scheduling processes. The aim is to streamline manufacturing operations, minimize downtime, and enhance operational efficiency.
These two fields of research are closely related, as in the scheduling of heavy-duty vehicles (HDV) in transportation, or on the contrary in the routing of Automated Guided Vehicles (AGVs) through machines in manufacturing.
Lastly, this work extends to the interplay between autonomous vehicles and manufacturing systems, to analyze how the implementation of platooning-based control techniques reflects within the sphere of manufacturing.
During the whole research, the objective is to uncover innovative insights and present pragmatic solutions for the aforementioned topics, while contributing to the ongoing discourse about technological advancements. The research conducted has the potential to significantly impact various industries and research fields, as the convergence of autonomous systems and manufacturing processes is navigated.

After a comprehensive analysis of the current state of the art for autonomous vehicles in Chapter 1, the first goal, developed throughout Chapter 2, is to enhance trajectory planning and control within predefined scenarios for platooning. This thesis primarily concentrates on the planning and automation of vehicle displacement over time, with minimal emphasis

on information exchange, and it assumes the absence of data retrieval failures. Various techniques are employed, and practical implementation is demonstrated using Wheeled Mobile Robots (WMRs). This implementation serves as a cost-effective testbed for the validation of new platooning algorithms, even in their early stages, with an acceptable level of reliability.
The second objective in Chapter 3 explores various scheduling algorithms to optimize production processes. These algorithms are not only presented but also applied in real production cells to showcase their efficiency in enhancing overall productivity.
Lastly, the third objective, treated in Chapter 4, involves the identification of potential platooning characteristics that apply to other domains, such as smart manufacturing systems. This exploration includes the inheritance and application of platooning-based control techniques to manufacturing systems to assess potential performance improvements. The interoperability between platoons and smart manufacturing systems is facilitated by the shared characteristics between platoon AVs and AGVs, similar to WMRs. While differences exist in their modeling, these shared traits enable the exploration of scenarios where clustering and WMR platoon formation benefit the entire infrastructure.
In summary, this work aims to assess the benefits of platooning techniques in autonomous vehicles and extends these findings to diverse domains, including smart manufacturing systems. The ultimate goal is the smooth integration of scheduling techniques with trajectory planning for platooning, creating a comprehensive framework for enhancing manufacturing system performance.

1.1 Autonomous vehicles fields of research

Autonomous vehicles (AVs) are a topic of significant interest in systems engineering. The vision of populating our roads exclusively with unmanned cars is one of the most ambitious projects of the century, pursued by numerous research centers worldwide. The transition from human-driven to self-driving vehicles necessitates the implementation of robust security protocols to ensure the safety of passengers and other vulnerable road users, such as pedestrians. Indeed, manned vehicles may exhibit unexpected behaviors that could potentially impact autonomous cars, leading to undesired actions.
Due to their complexity, AVs include various interconnected research fields that contribute to automating car movements. For instance, AVs must be resilient to cyber-attacks while ensuring passenger privacy throughout the entire journey. More broadly, security and safety are primary aspects of AVs, closely tied to the hardware and sensors integrated into the vehicle. However, the primary emphasis lies in achieving autonomous operations, particularly guidance and navigation on the road.
The road environment is inherently complex, demanding the consideration of numerous parameters when automating a car's movements. Smart sensors play a pivotal role in

capturing videos and images from the surroundings and transmitting them to the CPU for analysis and evaluation. Furthermore, vehicles can exchange their state information, enhancing the environment reconstruction process.
Once the environment is reconstructed, the automated system can plan a trajectory to follow overtime and provide accordingly a control action, by modifying the acceleration and steering of the vehicle, just like a human being would.
The Society of Automotive Engineers has defined six different levels of automation for unmanned vehicles [1], more precisely:

- **Level 0 (No automation)**: there is no automation and all driving tasks remain under the responsibility of the driver.
- **Level 1 (Hands-on)**: there is a driver assistance system that can handle steering, acceleration, and braking based on environmental data, but the vehicle still remains under the control of the driver.
- **Level 2 (Hands-off)**: the automated system can manage the vehicle's control, but the driver must actively monitor and be prepared to take control of the vehicle for decision-making processes.
- **Level 3 (Eyes-off)**: the dynamic driving tasks are managed by the automated system. However, some limitations in the driving modes may require the driver to take control of the vehicle's, within some seconds after a warning.
- **Level 4 (Mind-off)**: a high level of automation, able to perform all tasks in limited spatial areas or under specific traffic circumstances. The steering wheel might not even be required.
- **Level 5 (Full automation)**: the highest level of automation, able to manage all dynamic driving tasks in any situation, effectively transforming the driver into a passenger within the vehicle.

To have an indicator of the current state of autonomous vehicles, Tesla's latest autopilot system, a leader in the industry, falls somewhere between Level 2 and Level 3 automation [2]. It is evident that there is substantial progress needed to ensure the safety of fully driverless vehicles, especially when considering their potential interactions in real-world road environments. Literature has been studying AVs for decades, with a growing interest in the last 20 years, driven by significant technological advancements that have facilitated the development of new algorithms and methodologies for these vehicles.
Undoubtedly, many topics are associated with AVs, including:

- **Cybersecurity**:Ensuring the safety of passengers relies heavily on the software of AVs. Consequently, it is crucial to address vulnerabilities that make vehicles susceptible to cyberattacks by hackers who could potentially take control of the cars and make them perform undesirable actions. Numerous concerns regarding technological risks have been raised, as evidenced by several studies [3, 4, 5]. This theme is closely intertwined with user privacy, necessitating the establishment of a robust framework to prevent and recover from cyberattacks, such as the one proposed in[6] which makes use of deep learning algorithms.
- **Data fusion**: AVs gather data from various sensors placed on the vehicle, which must be integrated to provide the most accurate information about the vehicle's surroundings. For instance, [7] proposes a sensor fusion mechanism to combine 3D camera sensor data and Lidar sensor information. Similarly, [8] presents a real-time data fusion network equipped with fault diagnosis and fault tolerance mechanisms to recover from sensor failures.
- **Decision making**: Vehicles must make informed decisions based on the information collected and the current road conditions. Particularly in dense traffic situations, improper decision-making can lead to traffic deadlocks. For instance, [9] has developed a game-theoretic framework that enables agents to negotiate safely with other traffic participants. A subfield of this topic focuses on ethical decision-making, explored in greater detail in [10].
- **Information exchange**: This topic encloses all the possible communication channels between vehicles, such as Vehicle-To-Vehicle communication and Vehicle-To-Infrastructure, and even interactions with pedestrians who can contribute to information exchange through their smartphones [11, 12]. The union of these communication modes is referred to as Vehicle-to-Everything or when considering the entire infrastructure, the Internet of Vehicles, and it is anticipated that this will evolve into the Internet of Autonomous Vehicles [13].
- **Trajectory planning and control**: This topic, perhaps more than any other, aligns closely with systems engineering principles, as it deals with planning the vehicle's trajectory over time and guiding it along that path. Precise control of the vehicle is essential to closely follow the prescribed trajectory while adhering to physical constraints and traffic regulations. This field typically comprises two categories: longitudinal and lateral control.

1.1.1 Models for autonomous vehicles

Modeling serves as the initial step in applying control algorithms to unmanned vehicles. In the literature, numerous options exist, categorizing these models into two main groups:

kinematic models and dynamic models.
Kinematic models are typically employed in trajectory planning, where vehicle dynamics are often simplified or neglected to facilitate the rapid derivation of feasible paths. The widely used kinematic bicycle model, extensively discussed in [14], has been scrutinized for trajectory planning purposes. The study found that this model can be effectively employed for motion planning, provided that constraints on lateral acceleration are respected. Additional research in [15] supports the suitability of both the kinematic model and its dynamic variant for tracking reference trajectories, particularly at low and moderate speeds. During this phase, the vehicle's path is often analyzed concerning a predefined desired trajectory, without distinguishing between longitudinal and lateral displacement.
Conversely, for vehicle control applications, the model must be more precise, considering physical characteristics to offer a reliable representation of the vehicle. Tire-based models, such as Pacejka's model [16], have been developed to handle non-linear tire behavior effectively.
In some cases, models from other research fields can also be applied to autonomous vehicles. Lomonossoff's model [17], primarily designed for trains, can play a crucial role in accurately modeling a vehicle's longitudinal dynamics.
In summary, it is common practice to use two distinct models to represent vehicle dynamics: one is well-suited for trajectory planning, providing a simplified approximation of the vehicle's physical components, while the other offers a more detailed representation for actual displacement. The choice involves a trade-off between model simplicity, computational complexity, and fidelity to real-world vehicle behavior

1.1.2 Trajectory planning

Trajectory planning covers a broad area of research, particularly in the field of robotics and automation. Special attention is dedicated to determining feasible trajectories that can be safely pursued at high speeds, ensuring the safety of all involved agents.
In the literature, various algorithms have been developed, each with its own focus, such as minimizing time, energy consumption, or jerk over time. Some studies also investigate multi-objective functions that aim to strike a balance between these goals. For instance [18] introduced a Nonlinear MPC for urban traffic trajectory planning, even under zero-speed conditions, often challenging for modeling. They incorporated Pacejka's nonlinear tire model [19] and considered road boundaries as well as the presence of static and moving objects. The former, usually referred to as lane-keeping, has been a subject of study in many articles. For example, [20] employed a Convolutional Neural Network on raw image frames, while [21] developed an embedded system integrating lane detection and tracking using deep learning techniques. Additionally, [22] combined lane keeping with longitudinal speed control through MPC and Proportional Integral Derivative (PID) controllers to create a reliable system that minimizes lateral deviation while maintaining an acceptable

longitudinal speed.
On the other hand, obstacle avoidance plays a relevant role in ensuring passenger safety. [23] proposed a local planner to handle dynamic changes in the environment, while the global planner provides the fundamental reference waypoints to pursue in predefined scenarios.
arious approaches are used to generate optimal trajectories. For example, [24] creates clothoid tentacles based on the ego-centered reference frame of the vehicle, using an occupancy grid to classify each tentacle as navigable or not. The best tentacle is chosen as the reference trajectory based on criteria related to obstacle clearance, curvature change on the clothoid, and deviation from the initial obstacle-free trajectory. Similarly, [25] computes a feasible path considering kinematic constraints suitable for urban environments where low curvature tentacles provide greater passenger comfort.
In contrast, [26] employs quartic Bézier curves, a parametric curve type, to generate collision-free trajectories, balancing comfort and safety through optimization. [27] tested the Bézier curve path planner in a real rotary course scenario. Another approach involves B-Spline, a basis function containing a set of control points. B-Splines are widely used in autonomous driving, such as in [28], where they provide a robust controller for lane-change maneuvers while considering vehicle motion constraints and real-time requirements. Probabilistic models from other fields or research, such as biomedical [29], can help in improving the reliability of the controller.

1.1.3 Vehicle guidance and control

When it comes to controlling the vehicle, a common practice is to divide controllers into longitudinal and lateral components, as extensively reviewed in [30]. The longitudinal controller is responsible for regulating the vehicle's cruise velocity, while the lateral controller is tasked with steering the vehicle's wheels for path tracking and maintaining it on the centerline.
Longitudinal control faces a significant challenge due to non-linearity in the powertrain, leading to suboptimal performance at low speeds. To address this issue, [31] developed a two-level controller that incorporates the vehicle's reverse plant. The outer level determines the target speed based on the preceding vehicle on the road, while the inner level utilizes a Proportional-Integral (PI) controller to determine the appropriate accelerator pedal or brake percentage. An alternative approach is presented in [32], which proposes an adaptive solution based on a Model Reference Adaptive Controller. In general, pure pursuit path-tracking algorithms have proven effective for both mobile robots and autonomous vehicles. In this context, [33] introduced a variation called CF-Pursuit, which employs clothoid tentacles to generate the path. Adaptive pure pursuit, as seen in [34], optimizes lateral displacement for both high-curved and low-curved paths.
Lateral controllers primarily focus on keeping the vehicle in the middle of the lane. Given

that a car is a nonholonomic system, preventing sliding on the road is crucial. In [35] lateral controller is designed based on the Immersion and Invariance principle, achieving robust lateral tracking of a reference trajectory and smoother steering in simulations. Another approach, as outlined in [36], combines backstepping and sliding mode control, with stability verified through Lyapunov analysis and parameter tuning using Particle Swarm Optimization.

For lane-change maneuvers, some implementations integrate longitudinal and lateral controllers with Vehicle-to-Vehicle communication-based algorithms, as demonstrated in [37]. Similarly, [38] considers both lateral and longitudinal displacements within a neural, model-independent controller. Recent works [39, 40] consider also the implementation of navigation systems on cost-effective hardware to preliminary test the behavior of autonomous systems in indoor and outdoor environments.

1.2 Autonomous vehicle platooning: an overview

Vehicle platooning refers to a group of vehicles traveling closely together in a coordinated fashion. Platooning involves vehicles communicating with each other to retrieve information about their surroundings in order to virtually reconstruct the environment and consequently optimize their behavior on the road. Its primary application lies in the transportation industry, notably in freight transport, where multiple trucks can form platoons to enhance fuel efficiency and reduce operational costs. Autonomous vehicle platooning adds a degree of complexity to the system because it raises the need to optimally control a set of vehicles as a whole. Consequently, with respect to a single AV, more constraints have to be taken into account both in planning the leader/followers' trajectory and in their displacements. It is a subset of autonomous vehicles and more broadly of System of Systems (SoS) engineering, which involves multiple interconnected systems influencing each other's behavior. In the context of autonomous vehicles, SoS engineering pertains to coordinating multiple elements traveling the same path and interacting. Typically, a platoon comprises 3 to 20 vehicles, necessitating information exchange to enhance control reliability. In larger platoons, efficient information flow is essential, and it's established that all vehicles should have information about the leader for optimal coordination [41].

The advantages of platoon control are evident in their impact on road throughput. Platoon management can enhance stability and capacity within traffic flow, even in mixed traffic conditions [42]. However, inadequate platoon management can lead to road bottlenecks affecting the entire section.

Platoons can dynamically adjust their size by adding or removing vehicles based on traffic routing and their primary drawback lies in these dynamic changes, which may either slow down vehicles and reduce overall traffic flow or speed up some vehicles, potentially causing them to exceed speed limits.

However, vehicle platooning revealed to be very useful for trucks, as less space can be taken between two adjacent elements due to the faster reaction time, even if there still are concerns about how large platoons may impede traffic and about another vehicle trying to wedge itself between elements of the platoon. Lastly, the hardware to be mounted on each vehicle comprehends many sensors and thus it may be expensive in the near future. Overall, platooning represents a promising field of research in SoS engineering and many efforts have been made recently to enhance single autonomous vehicles, as well as fleets of AVs.
The distinction between single (independent) vehicles and platoons lies in their control mechanisms. Individual vehicles can select their control laws based on information and goals, whereas platoons are viewed as single entities from a traffic flow perspective (macroscopic) while being composed of multiple elements from a microscopic standpoint. This necessitates a dual-level controller approach: the high level treats the platoon as a single, extended vehicle within traffic, while the low level focuses on guiding individual vehicles to maintain inter-vehicle distances and ensure passenger safety.
This control process acts in two distinct stages: first, trajectory planning, which identifies a feasible and secure path for the vehicle to follow over time. Subsequently, the actual vehicle control occurs, with the aim of closely adhering to the desired path. Trajectory planning assumes a central role in autonomous vehicles, as it predicts vehicle movements along the road. In platooning, path planning primarily centers on the leader's trajectory but holds significance across various contexts, including applications within smart manufacturing systems. When dealing with platooning, and thus with SoS, there are usually two main control approaches: centralized and distributed. The former relies on a centralized architecture that retrieves all the data and handles the control of each element autonomously. Its main advantages are the simplicity both in creating the infrastructure and in the control law. The drawbacks, though, include the single point of failure and the non-scalability.
Distributed systems take the benefits of decentralization, adding communication between elements in order to provide a reliable and scalable control algorithm. Moreover, distribution in such systems boosts adaptability to different scenarios and ensures that the failure of an agent does not result in the failure of the whole system.
There are already several findings that show how autonomous driving could improve the throughput of the road, while reducing bottlenecks and fuel emissions, especially for platooning [43, 44]. In order for self-driving cars to overcome human drivers, they have to provide greater reliability, better driving performances, faster reaction time, and safer management of emergencies. Moreover, all these features must be stated in every condition, disregarding weather factors or unexpected events that may happen on the road. Moreover, in any urban environment, the vehicle will need to react safely to each type of unexpected event such as ill-behaved pedestrians, and pranksters [45].
Due to the rigid constraints that surround AVs in terms of safety and performance, it is clear that many approaches have been reviewed in the literature with the aim of finding the

best one for each possible scenario. The main challenge is to make these vehicles ever more performing and safer, thus guaranteeing a greater number of vehicles on the roads and a lower risk of accidents, ideally accident-prone. Usually, when dealing with platoons, great interest is put on longitudinal control and the respect of inter-vehicle distances, surely the key aspect of platooning. On the other hand, the lateral control is neglected, as the lane maintenance is kept employing a specific controller, such as the adaptive MPC in [46]. The planning and control modules may be overlapped, as in [47], where it has been designed an optimal control-based trajectory planning model that can be incorporated in platooning and with lane changing. For platooning, lane-change maneuver has been preliminary studied in [48], even if such maneuvers were revealed to be particularly heavy in terms of road congestion, especially for large platoons. Of course, in order to perform a lane-change maneuver, lateral control is required on each element of the platoon; this is handled with the sinusoidal controller explained in [49].
However, platoon research usually tackles the trajectory planning with regards to the leader, while followers reproduce the behaviour of a leading vehicle [50] separated by a safety inter-distance. They are constantly monitored utilizing safety zones, designed accordingly to road traffic rules and following principles introduced in [51] and [52]. The classification of safer or less safe areas leads to the choice of the trajectory generator algorithm to be used on followers.
Other works such as [53] consider cooperative driving to ensure that vehicles reach a predefined state of both position and velocity, employing Pontryagin's minimum principle to optimize longitudinal speed combined with MPC to avoid collisions. Platoons play a leading role especially in HDVs, as they are the class of vehicles that benefit the most from the characteristics of string line formation. In terms of fuel consumption reductions, as argued in [54], the air drag is drastically lowered, allowing up to 10% fuel reduction for trucks driving within the platoon. To this end, several works in the literature have been oriented at reducing the fuel consumption of trucks composing the platoons. In fact, [55] studied when it is efficient for an independent vehicle to drive faster and catch up with a platoon, in order to reduce long-term fuel consumption. In the same direction [56] evaluated the fuel-saving potential as offered by platooning under realistic conditions, while [57] designed a centralized truck platoon coordinator with dynamic vehicle plans that lead to reduced fuel consumption.
Other important aspects of HDV platooning are energy saving and enhanced transportation capacity, as pointed out in [58]. Experiments on three to four HDVs, which seems a reasonable number for platooning such type of vehicles, have shown the effectiveness in reducing energy consumption and CO_2 emissions with the aid of a lateral controller independent for each vehicle and a longitudinal one which maintains truck speed and clearance gap using Linear Quadratic (LQ) control. More in general, Linear Quadratic Regulator (LQR) seems suitable to handle trajectory planning, as the model considered is often linear and kinematic. Also, [59] uses it to obtain optimal control of a string of high-speed moving vehicles, with the more general theory provided by [60]. However, both works seem to be

ill-posed according to [61], which proposes a solution for detectability and stabilizability issues. Another approach consists of using Lyapunov control to guarantee string stability, necessary to prevent the propagation and amplification of any spacing error [62].
To mention that platooning represents a branch of SoS, and consequently it can present emergent behaviours due to the interconnection of the single subsystems (i.e. the element composing the platoon). The emergent properties of energy efficiency and transport efficiency have been studied in [63], which shows that the drivers' choice has a large impact on them. In particular, individual selfish/aggressive behaviours on road will lead to poor efficiency performances for everyone. Therefore, autonomous platooning can help in reducing the effects with a cooperation approach that enhances safety. In [64], for example, the usage of a local MPC extended by a collision-safety framework allows the achieving of a smaller inter-vehicle distances safely. This reflects in a higher road capacity, as platoons usually represent the bottleneck in traffic flow. In [65] control actions on platoons are computed with a prediction model with an overall faster decongestion of traffic.
Nonetheless, it is important to analyze autonomous platoons in conditions of mixed traffic, because the transition to unmanned vehicles will inevitably occur in a gradual fashion. In this direction, a MPC-based cooperative control has been developed in [66] to obtain traffic flow smoothness and stability, taking into consideration the unexpected behaviour of drivers within the road.

Topic	**References**
Single vehicle	[2, 20, 21, 22, 23, 24, 25, 26, 27, 28, 30] [31, 32, 33, 34, 35, 36, 37, 38, 46]
Platoon	[41, 42, 43, 44, 48, 54, 55, 56, 57, 58, 59, 61] [62, 63, 64, 65, 66]
Heavy-duty vehicles	[41, 43, 54, 55, 56, 57, 58, 62, 66]
Trajectory planning	[18, 23, 24, 25, 26, 27, 28, 47, 53]
Control	[20, 21, 22, 30, 31, 32, 33, 35, 36, 37, 38, 44] [46, 61, 65, 66]
Modeling	[14, 15, 16, 17, 19, 48]
Information exchange	[7, 8, 9, 11, 12, 13, 41]

Table 1.1: References for autonomous vehicles

1.3 Smart manufacturing systems: an overview

Another aim of this thesis is to investigate the potential enhancement of a smart manufacturing system's performance through the management of Automated Guided Vehicles (AGVs) trajectories and their scheduling through machines. In order to provide a more comprehensive analysis, Chapter 3 addresses product scheduling challenges in manufacturing machinery. Though this example may seem manufacturing-focused, a similar concept of scheduling exists in the transportation domain. Scheduling trucks within warehouses follows a comparable logic, on a larger spatial scale, to the scheduling of products through machines.
In both manufacturing and transportation domains, the core concept revolves around optimizing the allocation of resources to achieve efficient and effective operations. When it comes to manufacturing, this entails scheduling the production of various items on machines within a factory to minimize production time and resource utilization. On the contrary, in the transportation domain, a parallel need arises within warehouses, where the goal is to efficiently manage the flow of goods and materials. This involves scheduling the movements of trucks and other transport vehicles to ensure timely deliveries, minimize handling and storage costs, and streamline the logistics process. While the scale and specifics may differ, the fundamental principles of resource allocation and optimization remain a common thread between these two domains. Moreover, it represents the preliminary step preceding trajectory management and can be seamlessly integrated with it to address manufacturing issues entirely, considering both scheduling and the actual movement of AGVs within the system. The latter is treated in Chapter 4 with two comprehensive case studies.
During the production execution, it is well established that material flow management and material transport are fundamental processes for achieving optimal manufacturing operation performance. These production processes comprise how the material is transported and transferred within a manufacturing facility through several stages, from material extraction, product processing, recycling, and correspondent disposal [67]. The importance of searching for efficient management on these processes is that a lack of control negatively impacts the overall production performance [68]. Dynamic trajectory planning is a critical aspect of material flow and transportation in manufacturing. It involves computing real-time AGV motions from initial states to specific destinations, considering feasible paths and vehicle capabilities [69]. Contributions to this field can be categorized as centralized or decentralized approaches. In a centralized approach, as seen in [70], optimal control problems are used with differential-algebraic equations to describe vehicle characteristics and achieve near-optimal solutions. Conversely, decentralized approaches, such as the one highlighted in [71], enhance autonomous vehicle capabilities with path planning and motion coordination algorithms to ensure reliable path selection and conflict resolution. Additionally, hybrid approaches, like the one discussed in [72], combine trajectory planning and scheduling to minimize task completion times. Analogously, [73] employs a model

predictive planner to coordinate AGVs in a distirbuted fashion. Transport scheduling is the manufacturing procedure that sequences the tasks of each product to deliver it to the required destination while respecting constraints on its process and avoiding inefficiency, delays, congestion, and buffer usability [74]. This is usually referred to as short-term scheduling, whereas long-term scheduling manages the product demand over a longer time horizon (i.e., weekly or monthly), taking into consideration also the costs of inventory and employment [75]. Deep reinforcement learning was revealed to be promising too for short-term scheduling [76, 77], while in [78] Petri Nets and a heuristic based on artificial intelligence have been combined to solve flexible manufacturing systems (FMS) scheduling problems.
If all information is given a priori, offline scheduling can reduce the waiting time for products by planning the dispatching of products over machines before their arrival time.
While the state of the art in offline scheduling has provided valuable insights into optimizing tasks in predetermined conditions, the shift towards dynamic scheduling necessitates a focus on real-time adaptability and responsiveness to changing environments, as demonstrated in [79], with various reactive scheduling policies. Also link failures can be considered to guarantee the continuity of production [80, 81]. To address stochasticity, typically associated with processing times, [82] developed a multiobjective scheduling model. This model seeks to optimize product quality while minimizing tardiness, which refers to the delay or lateness in completing scheduled tasks.
In [83, 84] authors make use of genetic algorithms and, as for the offline scheduling, deep reinforcement learning is a suitable approach in case of the need for adaptability within the shop [85]. Similarly, [86] accomplishes a bi-objective optimization with the aid of reinforcement learning and a Mixed Integer Linear Programming (MILP) model. MILP and Constrained Programming (CP) are widely used to schedule jobs within facilities. It is the case of [87] which proposes a comparison between methodologies highlighting the advantages and drawbacks of both. Again, [88] makes use of a MILP model and memetic algorithm to solve the bi-objective cost function and minimize total setup time and the number of late jobs.
In modern manufacturing scheduling, digital twins can be employed to offer a virtual representation of physical manufacturing systems. These virtual replicas, often based on real-time data and simulation, enable manufacturers to optimize scheduling processes in several ways or to reconfigure the control of those systems [89]. Digital twin-driven scheduling has been deeply studied in recent years, with the overall architecture presented in [90], and has proven to be a valuable option to handle dynamic rescheduling necessary to adapt to both internal and external events [91]. Rescheduling in general helps in keeping competitive performance in manufacturing control [92]. Moreover, several digital twins can be aggregated, as in [93], to represent autonomous decision-making entities within the shop floor.
In summary, AGV trajectory planning and advanced real-time scheduling techniques offer an innovative approach to boost the performance and efficiency of smart manufacturing

systems. By addressing dynamic challenges in material flow and product scheduling, these methodologies pave the way for agile, responsive, and optimized manufacturing operations.

Topic	**References**
AGV	[69, 70, 71, 72, 73, 74]
Scheduling	[72, 74, 75, 76, 77, 78, 79, 82, 83, 84, 85, 86] [87, 88, 90, 91, 92, 93]
Manufacturing	[68, 78, 79, 80, 81, 83, 84, 85, 86, 89, 90, 92]

Table 1.2: References for smart manufacturing systems

Chapter 2

Platoon control in transportation

Autonomous vehicle platooning involves a group of self-driving vehicles traveling in a string formation, closely following each other. This concept has numerous advantages, making it an important solution to address several critical challenges in the transportation industry. The control of autonomous vehicle platooning allows to achieve numerous benefits, such as:

- **Reduced fuel emissions**: platooning can drastically reduce fuel consumption and, consequently, gas emissions.Vehicles traveling in a close formation experience reduced aerodynamic drag, which leads to improved fuel efficiency. Studies [94] have indicated that platooning can result in fuel savings of up to 10-20% for the entire platoon. The transportation sector is a major contributor to air pollution and climate change, thus implementing platooning can play a crucial role in achieving sustainability goals and combating climate change. Moreover, reduced fuel consumption directly translates to lower operational costs for companies, which can, in turn, lead to more competitive pricing for consumers.
- **Enhanced safety**: platooning can significantly improve road safety through precise coordination and reaction times of self-driving systems. The vehicles within a platoon can communicate and exchange information in real-time to keep a safe inter distance and speed, which reduces the risk of accidents caused by human error (estimated as 90% of road traffic accidents, with 57% of cases in which human error is the only cause [95]). Additionally, AVs can respond to potential hazards much faster than human drivers, further enhancing safety on the road.
- **Increased overall throughput of roads**: platooning allows for better utilization of road capacity, as vehicles are able to travel closely together and occupy less space on the road. This has a huge impact on traffic througput, especially on highways. By

minimizing the distance between vehicles and reducing traffic congestion, platooning helps optimize traffic flow, allowing more vehicles to traverse the same stretch of road in a shorter amount of time, enhancing overall transportation efficiency.

In conclusion, controlling efficiently autonomous vehicle platooning is fundamental to addressing critical challenges in the transportation sector. By reducing fuel emissions, enhancing road safety, and increasing the overall throughput of roads, autonomous vehicle platooning can offer a comprehensive and sustainable solution within road environment.

2.1 Modeling the dynamics of autonomous vehicles

Vehicle modeling is a foundational aspect of developing autonomous vehicles, serving as the mathematical representation that simulates a vehicle's behavior and dynamics. Modeling is a fundamental component of autonomous vehicle design, control, and decision-making processes. Through accurate vehicle modeling, engineers and researchers can better understand and predict how an autonomous vehicle will respond to various inputs and environmental conditions.
Vehicle modeling helps describe how a vehicle moves and behaves in response to different inputs, such as steering, throttle, and braking. Understanding a vehicle's dynamics is critical for creating control algorithms that ensure safe and precise navigation. Moreover, vehicle models are essential for designing control systems that stabilize the vehicle, maintain desired speeds, and execute complex maneuvers, such as lane changes and parking. Accurate modeling is crucial also for realistic and effective simulation, allowing engineers to assess the vehicle's performance under various conditions in a simulated environment before deploying the control algorithm on a real unmanned car.
Vehicle modeling can range from simple linear models to highly complex, physics-based representations, depending on the level of accuracy required for specific applications. Advanced models may account for factors like tire-road interaction, aerodynamics, suspension dynamics, and more.
In the following sections, discrete-time equivalents of the models are employed, even though continuous-time representations are possible. Additionally, each model pertains to an individual vehicle, with the platoon considered as an aggregation of these individual models. Effective communication and information exchange among vehicles is enhanced by the chosen controller type.

2.1.1 Kinematic model

The kinematic model represents one of the simplest yet useful models to reproduce vehicle's displacement along the road. It abstracts the vehicle's dynamics into a mathematical framework. There are primarily two types of kinematic models: first-order and second-order:

- **First-order kinematic model**: The vehicle's motion is described using basic parameters like position and orientation. It assumes that the vehicle can instantaneously change its velocity and steering angle. This model is simpler and more suitable for low-speed applications or when smooth and accurate control is not required. It doesn't consider the vehicle's acceleration directly. Its equations to update the position are:
$$\begin{cases} x_t = x_{t-1} + v_t \cos\theta_{t-1}\Delta t \\ y_t = y_{t-1} + v_t \sin\theta_{t-1}\Delta t \end{cases} \tag{2.1}$$
where x_t,y_t are the updated coordinates of the vehicle at time t and they are kept constant until instant $t+1$, v_t is the vehicle's velocity ad time t, θ_{t-1} its orientation at the previous instant and Δt the sampling time.
The equation for the heading update is:
$$\theta_t = \theta_{t-1} + \frac{v_t}{L}\delta\Delta t \tag{2.2}$$
where θ_t is the updated heading (orientation) of the vehicle at time t, L is the distance between the front and rear axles, and δ is the steering angle.
In general, the state variables are the position and orientation, while speed and steering angle and control variables.

- **Second-order kinematic model**: The second-order kinematic model is more sophisticated. It accounts for the vehicle's acceleration by including velocity and yaw rates. This model provides a more accurate representation of a vehicle's motion and is commonly used in higher-speed or more dynamic situations, where acceleration and deceleration play a significant role. Its equations are:
$$\begin{cases} x_t = x_{t-1} + \frac{v_t}{\dot\psi_t} \cdot \Big(\sin(\theta_{t-1} + \dot\psi_t\Delta t) - \sin(\theta_{t-1})\Big) \\ y_t = y_{t-1} + \frac{v_t}{\dot\psi_t} \cdot \Big(\cos(\theta_{t-1}) - \cos(\theta_{t-1} + \dot\psi_t\Delta t)\Big) \\ v_t = v_{t-1} + a\Delta t \\ \theta_t = \theta_{t-1} + \dot\psi_t\Delta t \end{cases} \tag{2.3}$$
where ψ is the rate of change of heading, namely yaw rate, and a is the vehicle's acceleration.

In this model, the position, heading, and speed are state variables, while yaw rate and acceleration are used as control variables.
In literature a simplification of the second-order model is employed when dealing with one-dimensional trajectories:

$$\begin{cases} x_t = x_{t-1} + v_{t-1}\Delta t + \frac{1}{2}a_{t-1}\Delta t^2 \\ v_t = v_{t-1} + a_{t-1}\Delta t \end{cases} \tag{2.4}$$

In summary, the main difference between first-order and second-order kinematic models for autonomous vehicles is the level of detail they provide regarding the vehicle's motion. The second-order model, being more comprehensive, is better suited for situations where precise control and handling dynamics are essential, while the first-order model offers a simpler approximation appropriate for more straightforward scenarios.

2.1.2 Lomonossoff's model

A more sophisticated model for a one-dimensional path can be retrieved from train literature and it is called Lomonossoff's model, also exploited in [96]. The model has the advantage of incorporating not only the vehicle's mass but also various physical parameters, including aerodynamic and mechanical resistances. The equations describing the motion are well-suited for HDVs due to their similarities in dynamics with trains:

$$\begin{cases} \dot{x}(t) = v(t) \\ W'\dot{v}(t) = f(t) - (C^a + C^b v(t) + C^c v^2(t)) - Wg\sin\alpha(t) \end{cases} \tag{2.5}$$

where x and v are state variables for position and speed of the vehicle, f is the control input, corresponding to the tractive effort and it is expressed in kN, C^a, C^b, C^c are the Davis constants, related respectively to mechanical resistance, viscous mechanical resistance and aerodynamic resistance, W is the vehicle's tare mass and W' is its effective mass, including rotary allowance, both expressed in tonnes. The slope angle of the road, α, is generally neglected.
The dynamics of the longitudinal motion are represented by prompting a tractive effort, with a maximum value similar to what happens in train modeling.

2.1.2.1 Linearization of the Lomonossoff's model

The model presented in Eq. 2.5 is clearly nonlinear. It can be linearized for each planned instant t_p around a working state/control couple $(\bar{v}, \bar{f})$, supposing no acceleration in that instant of time.

The resulting linear approximation that represents the evolution of the system over time is:

$$\delta\dot{x} = A_p \delta x + B_p \delta f \tag{2.6}$$

where:

$$\delta x = [x(t) - \bar{x}(t) \quad v(t) - \bar{v}(t)]^T, \quad \delta f = \left[f(t) - \bar{f}(t)\right] \tag{2.7}$$

$$A_p = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{C^b + 2C^c \bar{v}(t)}{W'} \end{bmatrix}, \quad B_p = \begin{bmatrix} 0 \\ 1/W' \end{bmatrix} \tag{2.8}$$

It has to be noted that the A_p matrix is time-variant, as it depends on the actual speed of the vehicle.

2.1.3 Micro-Macro METANET model

Autonomous vehicles and platoons can also be viewed from a broader perspective, specifically in the context of road traffic. To achieve this, the traffic conditions and the presence of platoons within the mainstream can be modeled using the Micro-Macro METANET (M^3-net) model, introduced in [97] and derived from the well-known METANET model [98].
The Micro-Macro METANET model, for sake of simplicity, is reported in the following referring to the single stretch. In this version of the M^3-net model the freeway stretch is divided into N sections, denoted with $i = 1, \dots, N$, each of length L_i, and with λ_i lanes. Even the stretch version of the model is discretized in time, specifically the time horizon is divided into K time steps, indicated hereafter with $k = 0, \dots, K$, and with sample time interval T. With regard to platoon presence in the model, let Z represent the total number of platoons, and let z, with $z = 1, \dots, Z$, denote a generic platoon. It is important to note that in this model, platoons are represented in an aggregate manner, as single long vehicles with specified occupancy.
The dynamic evolution of the M^3-net model is described by means of aggregate quantities, referring to the whole traffic stream, and microscopic variables describing instead the movements of platoons in the freeway. The quantities related to the macroscopic behavior are:

- $\rho_i(k)$, the traffic density in section i at time instant kT (expressed in vehicles per kilometre per lane);
- $\bar{\rho}_i(k)$, the traffic density including the presence of platoons in section i at time instant kT (expressed in vehicles per kilometer per lane);
- $v_i(k)$, the mean traffic speed in section i at time instant kT (expressed in kilometres per hour);

- $q_i(k)$, the traffic volume which leaves section i during time interval $[kT, (k+1)T)$ (expressed in vehicles per hour);

whereas the macroscopic quantities referred to the on-ramps and off-ramps are:

- $l_i(k)$, the queue length of vehicles waiting on the on-ramp of section i at time instant kT (expressed in vehicles);
- $b_i(k)$, the queue length of platoons waiting on the on-ramp of section i at time instant kT (expressed in vehicles);
- $d_i(k)$, the traffic volume requiring to access section i from the on-ramp during time interval $[kT, (k+1)T)$ (expressed in vehicles per hour);
- $r_i(k)$, the on-ramp traffic volume entering section i during time interval $[kT, (k+1)T)$ (expressed in vehicles per hour);
- $s_i(k)$, the off-ramp traffic volume exiting section i during time interval $[kT, (k+1)T)$ (expressed in vehicles per hour).

Some macroscopic variables are introduced to represent the inflows from the mainstream:

- $l_0(k)$, the queue length of vehicles waiting in the mainstream to enter section 1 at time kT (expressed in vehicles);
- $b_0(k)$, the queue length of platoons waiting in the mainstream to enter section 1 at time kT (expressed in vehicles);
- $d_0(k)$, the traffic volume requiring to access section 1 from the mainstream during time interval $[kT, (k+1)T)$ (expressed in vehicles per hour);
- $q_0(k)$, the traffic volume entering section 1 from the mainstream during time interval $[kT, (k+1)T)$ (expressed in vehicles per hour).

Finally, the microscopic variables introduced to represent the movements of platoons in the freeway are:

- $v^z(k)$ is the speed of platoon z at time instant kT (expressed in kilometres per hour);
- $p^z(k)$ is the distance covered by platoon z at time instant kT (expressed in kilometres);

The model has been designed to track the presence of platoons on the freeway in order to understand what traffic conditions they will encounter on their route and how to adjust their speed accordingly. To describe the behavior of platoons in the mainstream, for each platoon z and for each time step k, the position of a generic platoon is updated as:

$$p^z(k+1) = p^z(k) + v^z(k)T \tag{2.9}$$

In absence of control, it is assumed that the speed of each platoon $v^z(k)$ cannot overcome the maximum speed allowed for platoon z, i.e. $v^{z,\max}$, nor the mean speed in the section in which the platoon is traveling. To this end, $i^z(k)$ denotes the section index that corresponds to the position of the platoon z at time step k. This index is updated based on the distance covered $p^z(k)$. The average speed of the traffic flow that platoon z encounters at time step k is denoted by $v_{i^z(k)}(k)$, while the speed of each platoon in the absence of control is given by:

$$v^z(k) = \min\left\{v^{z,\max}, v_{i^z(k)}(k)\right\} \tag{2.10}$$

In the M^3-net model the aggregate variables representing the traffic behavior in the mainstream need to be updated to take into account the presence of platoons. As in the METANET model, the conservation equation referring to the traffic density in which platoons are not included, for $i = 1, \dots, N$, $k = 0, \dots, K-1$, is given by:

$$\rho_i(k+1) = \rho_i(k) + \frac{T}{L_i \lambda_i}\Big[q_{i-1}(k) - q_i(k) + r_i(k) - s_i(k)\Big] \tag{2.11}$$

The traffic density including platoons is based on Eq. 2.11 and defined as:

$$\bar{\rho}_i(k) = \rho_i(k) + \sum_{z=1}^{Z} \gamma_i^z(k) o^z \tag{2.12}$$

in which o^z is the occupancy corresponding to the z-th platoon (expressed in [veh/km/lane]), while $\gamma_i^z(k)$ is a binary variable adopted to indicate the presence, at time step k, of platoon z in section i. This binary variable is defined as:

$$\gamma_i^z(k) = \begin{cases} 1 & \text{if } i^z(k) = i \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

Also, the speed dynamics for each section i and for each time step k is computed by taking into account the presence of platoons and hence by considering the density $\bar{\rho}_i(k)$, i.e.

$$\begin{aligned} v_i(k+1) = v_i(k) &+ \frac{T}{\tau}\Big[V_i(k) - v_i(k)\Big] + \frac{T}{L_i} v_i(k)\Big[v_{i-1}(k) - v_i(k)\Big] \\ &- \frac{\nu T[\bar{\rho}_{i+1}(k) - \bar{\rho}_i(k)]}{\tau L_i[\bar{\rho}_i(k) + \chi]} - \Delta T \frac{v_i(k) r_i(k)}{L_i[\bar{\rho}_i(k) + \chi]} \end{aligned} \tag{2.14}$$

where τ, ν, χ, and Δ are model parameters, while the steady-state speed-density relation is given by:

$$V\left(\bar{\rho}_i(k)\right) = v_i^{\mathrm{f}} \cdot \left[1 - \left(\frac{\bar{\rho}_i(k)}{\rho_i^{\max}}\right)^l\right]^m \tag{2.15}$$

in which $\rho_i^{\max}$ is the jam density [veh/km/lane], v_i^{f} is the free-flow speed [km/h], and l, m are other model parameters.
Note that the traffic flow to be used in Eq. 2.11 is computed as $q_i(k) = \rho_i(k)v_i(k)\lambda_i$. It is assumed that platoons can enter the stretch from on-ramps or through the mainstream. In addition, we assume that the expected arrival of a platoon z at an on-ramp is known and modeled through a binary constant $a_i^z(k)$ that, for each time step k, is defined as:

$$a_i^z(k) = \begin{cases} 1 & \text{if platoon } z \text{ arrives at the on-ramp } i \\ 0 & \text{otherwise} \end{cases} \tag{2.16}$$

Similarly, the arrival of a generic platoon z to the mainstream is modeled using the binary constant $a_0^z(k)$, which is defined in a manner analogous to Eq. 2.16. These binary constants are defined to ensure that a platoon z cannot simultaneously enter from both the on-ramps and the mainstream. The effective entrance of a generic platoon z from an on-ramp onto the freeway at time step k is represented by the binary variable $y_i^z(k)$:

$$y_i^z(k) = \begin{cases} 1 & \text{if platoon } z \text{ enters from on-ramp } i \\ 0 & \text{otherwise} \end{cases} \tag{2.17}$$

The platoon's entry from the mainstream at time step k is modeled with the variable $y_0^z(k)$. Given the schedule of platoon arrivals at entry ramp i, platoon z enters the freeway, i.e., $y_i^z(k) = 1$, if and only if the following conditions are satisfied:

$$r_i(k) + \frac{1}{T}\left[a_i^z(k)n^z + b_i^z(k)\right] \le r_i^{\max} \tag{2.18}$$

$$r_i(k) + \frac{1}{T}\left[a_i^z(k)n^z + b_i^z(k)\right] \le r_i^{\max} \cdot \frac{\rho_1^{\max} - \bar{\rho}_1(k)}{\rho_1^{\max} - \rho_1^{\mathrm{cr}}} \tag{2.19}$$

while a generic platoon z enters from the mainstream, i.e. $y_0^z(k) = 1$, if the following conditions are both satisfied:

$$q_0(k) + \frac{1}{T}\left[a_0^z(k)n^z + b_0^z(k)\right] \le q_0^{\max} \tag{2.20}$$

$$q_0(k) + \frac{1}{T}\left[a_0^z(k)n^z + b_0^z(k)\right] \le q_0^{\max} \cdot \frac{\rho_1^{\max} - \bar{\rho}_1(k)}{\rho_1^{\max} - \rho_1^{\mathrm{cr}}} \tag{2.21}$$

where $r_i^{\max}$ is the maximum flow that can enter the on-ramp i, $q_0^{\max}$ is the maximum flow that can enter the mainstream, and n^z is the number of trucks that constitute platoon z.

In particular, $b_i^z(k)$ and $b_0^z(k)$ are virtual buffers defined for each platoon z whose length can be 0 or equal to the number of vehicles n^z that compose the platoon. The dynamic evolution of these virtual buffers are given by:

$$b_i^z(k+1) = b_i^z(k) + a_i^z(k)n^z - y_i^z(k)n^z \tag{2.22}$$

$$b_0^z(k+1) = b_0^z(k) + a_0^z(k)n^z - y_0^z(k)n^z \tag{2.23}$$

If conditions specified in Eqs. 2.18-2.19 and Eqs. 2.20-2.21 are not met, the corresponding binary variables $y_i^z(k)$ and $y_0^z(k)$ are assigned a value of 0, ensuring that buffer lengths, $b_i^z(k)$ and $b_0^z(k)$, do not decrease to 0. This prevents the processing of the subsequent platoon $z+1$ until the former conditions are met, if platoon z fails to enter from the on-ramps, or until the latter conditions are satisfied if platoon z fails to enter from the mainstream. Under these circumstances, the virtual queues are respecitvely updated as follows:

$$b_i(k) = \sum_{z=1}^{Z} b_i^z(k) \tag{2.24}$$

$$b_0(k) = \sum_{z=1}^{Z} b_0^z(k) \tag{2.25}$$

The queue length at on-ramps and at mainstream are computed as

$$l_i(k+1) = l_i(k) + T[d_i(k) - r_i(k)] \tag{2.26}$$

$$l_0(k+1) = l_0(k) + T[d_0(k) - q_0(k)] \tag{2.27}$$

where the on-ramp entering flow is obtained as

$$r_i(k) = \min\left\{ d_i(k) + \frac{l_i(k)}{T}, r_i^{\max}, r_i^{\max} \cdot \frac{\rho_i^{\max} - \bar{\rho}_i(k)}{\rho_i^{\max} - \rho_i^{\mathrm{cr}}} \right\} \tag{2.28}$$

while the traffic flow entering the first section from the mainstream is given by

$$q_0(k) = \min\left\{ d_0(k) + \frac{l_0(k)}{T}, q_0^{\max}, q_0^{\max} \cdot \frac{\rho_1^{\max} - \bar{\rho}_1(k)}{\rho_1^{\max} - \rho_1^{\mathrm{cr}}} \right\} \tag{2.29}$$

In summary, the $M^3 - net$ model serves as a valuable tool in traffic modeling when the objective is to capture and analyze traffic conditions at a macroscopic level. Moreover, by incorporating the microscopic dynamics of platoons, the $M^3 - net$ model enables a more comprehensive understanding of how platoons influence traffic dynamics.

2.2 Controlling the displacement of autonomous vehicles

There exist numerous control techniques available for planning trajectories or executing the actual motion of autonomous vehicles on the road. In the context of platooning, the techniques applied to individual vehicles are often complemented by distributed techniques to facilitate communication among platoon elements.
The choice of one technique over another may depend on several factors, including:

- The type of model, with a clear distinction between control techniques for linear and nonlinear models;
- The computational speed required;
- The level of accuracy needed;
- And more.

In the following, some of the control techniques widely used in the platoon domain, and especially in this thesis, are listed.

2.2.1 Linear Matrix Inequalities

Linear Matrix Inequalities (LMIs) offer a mathematical framework to address key control and optimization challenges. In the context of platooning, where a group of autonomous vehicles follows a lead vehicle closely and cooperatively, LMIs are a valuable tool for ensuring safe and efficient operation. A standard form for LMIs is:

$$F(\underline{x}) := F_0 + \sum_{i=1}^{m} x_i F_i > 0 \tag{2.30}$$

where $x \in \mathbb{R}^m$ is a vector of real numbers, namely the decision variables, and the matrices $F_i \in \mathbb{R}^{n \times n}, i = 0, ..., m$, are given and assumed symmetric. The non-strict relaxation allows for $F(\underline{x}) \geq 0$
Solving an LMI means finding a set of vectors $\underline{x}$ such that $F(\underline{x}) > 0$.
Moreover, a set of LMIs $F_1(\underline{x}) \geq 0$, $F_2(\underline{x}) \geq 0$,...,$F_k(\underline{x}) \geq 0$ can be represented as one single LMI:

$$F(\underline{x}) = \begin{bmatrix} F_1(\underline{x}) & & & \\ & F_2(\underline{x}) & & \\ & & \ddots & \\ & & & F_k(\underline{x}) \end{bmatrix} \tag{2.31}$$

This equation is beneficial in control theory because LMI's arise as functions of matrix variables rather than scalar-valued decision variables

2.2.1.1 Application of LMI to minimax team decision problems

Minimax team decision problems are a class of decision-making situations involving teams or groups of individuals working together to make uncertain choices. Robustness is one of the key arguments of minimax decision problems. Robustness refers to the ability of a decision or strategy to maintain acceptable performance even when faced with the most unfavorable or adverse conditions and it is indeed a fundamental concept in minimax decision problems. In this problem, each player has limited information that could differ from the other players in the team. To prove this, the following team decision problem can be considered:

$$\inf_{\mu} \sup_{0 \neq \underline{x} \in \mathbb{R}^n} \frac{J(\underline{x}, \underline{u})}{\|\underline{x}\|^2} \tag{2.32}$$

subject to:

$$\begin{cases} y_i = C_i \underline{x} \text{ for } i = 1, ..., N \\ u_i = \mu_i(y_i) \text{ for } i = 1, ..., N \end{cases} \tag{2.33}$$

where $u_i \in \mathbb{R}^{m_i}$, $m = m_1 + ... + m_N$, $C_i \in \mathbb{R}^{m_i \times n}$, for $i = 1, ..., N$.
$J(\underline{x}, \underline{u})$ can be a quadratic cost given by:

$$J(\underline{x}, \underline{u}) = \begin{pmatrix} \underline{x} \\ \underline{u} \end{pmatrix}^T \begin{pmatrix} Q_{xx} & Q_{xu} \\ Q_{ux} & Q_{uu} \end{pmatrix} \begin{pmatrix} \underline{x} \\ \underline{u} \end{pmatrix} \tag{2.34}$$

with Q_{uu} positive definite ($Quu > 0$).
The players $u_1, ..., u_N$ make up a team, which plays against nature, represented by the vector x, using μ:

$$\mu(Cx) = \begin{pmatrix} \mu_1(C_1 x) \\ \vdots \\ \mu_N(C_N x) \end{pmatrix} \tag{2.35}$$

The game is formulated as a minimax problem, where the team is the minimizer and the nature is the maximizer, and [99] proven that if there is a solution to the static minimax team problem, then linear decisions are optimal and a linear optimal solution

$$\mu_i(y_i) = K_i y_i \tag{2.36}$$

can be found by solving a LMI.

2.2.2 Linear Quadratic Regulators

Linear Quadratic Regulator (LQR) and Linear Quadratic Tracking (LQT) are fundamental control techniques used in the field of control systems engineering. They are employed to optimize the performance of systems characterized by a linear time-invariant plant, to optimize a quadratic cost function.
LQR is a control strategy designed to minimize a cost function that quantifies the system's performance. It achieves this by adjusting the control inputs in a way that optimizes the system's response, taking into account both the desired behavior and the system's inherent dynamics. LQR is particularly well-suited for stabilizing unstable systems.
The LQR problem can be mathematically defined as follows: Consider a continuous-time linear time-invariant system represented by the state-space equations:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.37}$$

where $\dot{x}$ is the evolution of the state vector over time, representing the system's dynamics, $u(t)$ is the control input vector, and A and B are system matrices.
The goal of LQR is to design a control law $u(t)$ that minimizes a cost function J over an infinite time horizon:

$$J = \int_0^{\inf} \left[x^T(t)Qx(t) + u^T(t)Ru(t) \right] dt \tag{2.38}$$

with Q and R respectively the state weighting and input weighting matrices. They represent the importance of tracking different states and the cost of control effort and they can be time-variant, namely $Q(t)$ and $R(t)$.
The optimal control law for the LQR problem is obtained by solving the continuous-time algebraic Riccati equation (CARE):

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{2.39}$$

where P is the state feedback gain matrix and it is positive definite.
Finally, the optimal control $u^*(t)$ is computed as:

$$u^*(t) = -R^{-1}B^T Px(t) \tag{2.40}$$

An equivalent discrete-time version can be obtained by the system's discretization and by solving the discrete-time algebraic Riccati equation (DARE) to find the matrix P.
Additionally, there exists an equivalent formulation over a finite time horizon, suitable for applications where the control objectives are primarily concerned with achieving specific performance goals within a defined time frame. More in detail, Eqs. 2.38 and 2.39 become:

$$\begin{aligned} J &= x^T(t_1)F(t_1)x(t_1) + \int_{t_0}^{t_1} \left[x^T(t)Qx(t) + u^T(t)Ru(t) \right] dt \\ A^T P(t) &+ P(t)A - P(t)BR^{-1}B^T P(t) + Q = -\dot{P}(t) \end{aligned} \tag{2.41}$$

with the boundary condition $P(t_1) = F(t_1)$.
LQR provides an optimal control law that minimizes a quadratic cost function, offering the best possible control strategy for linear time-invariant systems under the specified cost criteria. Although it can be extended to address time-varying systems, LQR may not be suitable for highly nonlinear systems without the use of linearization techniques, which can potentially degrade performance. Furthermore, to address system model inaccuracies and uncertainties, an extended controller, known as the Linear Quadratic Gaussian (LQG) regulator, incorporates a Kalman filter for state estimation, accounting for potential disturbances both in the state and its measurements.
LQT is an extension of LQR, tailored for applications involving trajectory tracking. While LQR focuses on optimizing the system's response to a fixed setpoint, LQT goes a step further by ensuring that the system can accurately follow a specified trajectory, namely x_r, over time. It accomplishes this by incorporating a tracking error term into the cost function, penalizing deviations from the desired trajectory. Indeed, LQT aims to minimize a cost function J that quantifies the tracking error over a finite time horizon $[0, T]$. In the following, the discrete-time LQT approach is discussed, with the cost function formulated as:

$$\begin{aligned} J = {} & (x(T) - x_r(T))^T \, Q_T \, (x(T) - x_r(T)) + \\ & + \sum_{k=0}^{T-1} \left[(x(k) - x_r(k))^T \, Q \, (x(k) - x_r(k)) + u(k)^T R u(k) \right] \end{aligned} \tag{2.42}$$

with the term outside the summation necessary to consider the update of the state at the final instant, without control action. For this reason, usually $Q_T \neq Q$. To find the optimal control, the DARE must be solved:

$$P(t) = A^T P(t+1)[\mathbb{1} + BR^{-1}BP(t+1)]^{-1}A + Q \tag{2.43}$$

And subsequently the vector difference equation:

$$g(t) = A^T \left\{ \mathbb{1} - \left[P^{-1}(t+1) + BR^{-1}B\right]^{-1} BR^{-1}B \right\} g(t+1) + Qx_r(t) \tag{2.44}$$

Solving for the optimal state $x^*(t)$ as:

$$\begin{aligned} x^*(t+1) &= (A - BL(t))x^*(t) + BL_g(t)g(t+1) \\ L(t) &= \left[R + B^T P(t+1)B\right]^{-1} BP(t+1)A \\ L_g(t) &= \left[R + B^T P(t+1)B\right]^{-1} B^T \end{aligned} \tag{2.45}$$

where L and L_g are computed based on the system dynamics.
Lastly, the resulting optimal control is:

$$u^*(t) = -L(t)x^*(t) + L_g(t)g(t+1) \tag{2.46}$$

2.2.3 Proportional-Integral-Derivative controller

The Proportional-Integral-Derivative (PID) controller is one of the most widely used control mechanisms in the field of engineering and automation. It is a feedback controller that plays a crucial role in regulating and stabilizing a huge variety of dynamic processes. The PID controller is valued for its simplicity, effectiveness, and versatility, making it a fundamental component in various industries and applications. It consists of three terms, even if for some systems only one or two terms are used:

- **Proportional term**: The proportional term produces an output signal that is directly proportional to the current error. It measures the difference between the desired setpoint and the actual process variable. The proportional action responds to the present error and provides immediate corrective action. The proportional gain (K_p in the following) determines the strength of this response. Proportional-only controllers exhibit a significant limitation in nullifying the steady-state error. Thus, they keep a persistent error, known as offset, which cannot be eliminated by exploiting only proportional controllers.
- **Integral term**: The integral term accumulates past errors over time, relying on both the error magnitude and the duration it persists within the system. For this reason, its control action is small at the beginning and it increases over time, bringing the error to zero slowly at the beginning and brutally at the end. In fact, a pure integral controller can potentially overshoot the zero-error state as it diligently eliminates any persistent error. The integral action is critical for systems with constant disturbances or biases and can accelerate the movement of the process toward the setpoint. The integral gain (K_i) governs the aggressiveness of this correction.
- **Derivative term**: The derivative term anticipates the future error by assessing the rate of change of the error signal. It dampens abrupt changes in the process variable, enhancing system stability and reducing overshoot. Even if the most used combination is the PI controller, the derivative term usually takes action when performance need improvements in settling time and overall stability. The derivative gain (K_d) controls the extent of this dampening.

Thus, the resulting control action is:

$$u(t) = K_p e(t) + K_i \int_0^T e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{2.47}$$

where $e(t)$ is the error between the outuput and the desired behaviour of the system. Adjusting the control parameters is essential for optimizing system performance. While stability remains a fundamental requirement, varying the gain values can influence aspects such as settling time and overshoot. PID tuning, although conceptually intuitive, can be challenging when multiple conflicting objectives need to be met. Tuning can be done manually using various methods, such as the Ziegler-Nichols [100] or automatically with the assistance of different tools.

2.2.4 Event-based controllers

In the context of autonomous driving, event-based controllers can be used for real-time decision-making based on sensor inputs and predefined triggers or events. These controllers operate by responding to specific events or conditions encountered during the vehicle's operation, guiding it through various scenarios safely and efficiently.
The flowchart representation of an event-based controller for autonomous driving would incorporate decision nodes and branches that respond to events such as obstacle detection, lane departures, traffic signals, and more. Each event triggers a corresponding action or sequence of actions to ensure the vehicle's appropriate response. The equations and logic for an event-based controller can be complex and system-specific, as they depend on the specific events and conditions being monitored and controlled. However, some general components and equations that may be relevant are:

- **Event Detection**: Event detection logic involves identifying specific conditions or events that are critical for safe autonomous driving. These conditions could include obstacles in the vehicle's path, sudden changes in road conditions, traffic signals, or other vehicles' behavior. It relies on information provided by sensors about the surroundings
- **Control Action:** Once an event is detected, the controller determines the appropriate control actions. This step is usually left to more complex controllers capable of incorporating the vehicle's dynamics and its accurate modeling to supply the optimal control action.
- **Event Handling and Reset Logic**: After taking action in response to an event, the controller needs to monitor whether the event condition persists. If the condition is no longer met, the event flag should be reset.

In essence, the flowchart embodies the logic and decision-making processes that enable an autonomous vehicle to navigate complex environments, but to handle the actual vehicle's displacement they are usually paired with the aforementioned controllers. Thus, the event-based controller is responsible for making the decision, while other controllers, handling the vehicle's dynamics, are in charge of executing the control actions to ensure accurate movement on the road.

2.3 Robust control in vehicles platooning through LMI

To first assess the problem of robust control in vehicle platooning, a one-dimensional scenario is designed. Each element's trajectory is modeled using the first-order kinematic model of Eq. 2.1. In this scenario, the heading update is neglected, the y-coordinate is unnecessary, and noise is introduced into the system. Consequently, the resulting equation for discretized displacement is:

$$x_i(t+1) = x_i(t) + v_i(t)\Delta t + w_i(t) \qquad t = 0, ..., T-1 \tag{2.48}$$

where x_i is the position of the i^{th} vehicle, v_i its velocity, w_i the possible disturbance and Δt the sampling time. This simple approach ensures a fast modification of the conditions and paired with rate limiters it can provide the bounding of the input respecting physical constraints.
In matrix form, the platoon of M vehicles is the concatenation of each element's model and may be formalized as follows:

$$\underline{x}(t+1) = A\underline{x}(t) + \Delta t B\underline{v}(t) + \underline{w}(t) \quad t = 0, ..., T-1 \tag{2.49}$$

with A and B identical matrices $\in \mathbb{R}^{M\times M}$.
In addition, it is assumed that each vehicle can access the information about its own position and on both the preceding and the following vehicle positions:

$$y_i = C_i x_i \tag{2.50}$$

and for the generic i^{th} vehicle:

$$C_i = \begin{bmatrix} 0 & ... & 0 & 1 & 0 & 0 & 0 & ... & 0 \\ 0 & ... & 0 & 0 & 1 & 0 & 0 & ... & 0 \\ 0 & ... & 0 & 0 & 0 & 1 & 0 & ... & 0 \end{bmatrix} \tag{2.51}$$

where $C_i \in \mathbb{R}^{3\times M}$.
The first and last vehicles of the platoon have slightly different $C_i \in \mathbb{R}^{2\times M}$ matrix:

$$C_1 = \begin{bmatrix} 1 & 0 & ... & 0 \\ 0 & 1 & ... & 0 \end{bmatrix} \quad C_M = \begin{bmatrix} 0 & ... & 1 & 0 \\ 0 & ... & 0 & 1 \end{bmatrix} \tag{2.52}$$

The structure of these matrices ensures that each vehicle monitors its own position, as well as the positions of the preceding and following vehicles in the formation. The trajectory of the i^{th} vehicle is adjusted with respect to the predefined reference path, denoted as $x_i^d(t)$. For the sake of notation, this adjustment involves the following change of variables:

$$\begin{cases} \tilde{x}_i(t) = x_i(t) - x_i^d(t) \\ \tilde{v}_i(t) = v_i(t) - v_i^d(t) \end{cases} \tag{2.53}$$

The problem can be defined as a minimax problem analogous to the one explained in Section 2.2.1.1:

$$\inf_{\underline{v}} \sup_{\underline{w} \neq 0} \frac{J(\underline{v}, \underline{w})}{||\underline{w}||^2} \tag{2.54}$$

subject to Eq. 2.49 where the cost function J is designed as:

$$J(\underline{v}, \underline{w}) = \sum_{i=1}^{M} \sum_{t=0}^{T-1} \alpha_i \tilde{x}_i^2(t) + \gamma_i \tilde{v}_i^2(t) + \sum_{i=2}^{M} \sum_{t=0}^{T-1} \beta_i (\tilde{x}_i(t) - \tilde{x}_{i-1}(t))^2 \tag{2.55}$$

In Eq. 2.55 α_i,β_i and γ_i represent gains that give primary importance respectively to tracking the trajectory, restoring the optimal inter-distance, and minimizing the difference from the desired velocity.

Theorem 1. Let's consider a time horizon of two intervals, i.e. $t = 0, ...T$ and $T = 1$. The trajectory of the platoon described by $\underline{x}(t)$ can be modified in real-time, according to the problem defined by Eqs. 2.54 and 2.49 by an optimal control law $v(t) = Kx(t)$, which is linear, where K is the solution of the following LMI:

$$\min_{\theta, K} \theta \tag{2.56}$$

s.t.

$$K = \begin{bmatrix} k_{1,1} & k_{1,2} & 0 & 0 & 0 & ... & 0 & 0 \\ 0 & 0 & k_{2,1} & k_{2,2} & k_{2,3} & ... & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & 0 & 0 & ... & k_{M,M-1} & k_{M,M} \end{bmatrix} \tag{2.57}$$

$$\begin{pmatrix} \hat{\mathbf{Q}}_{xx} - \theta \mathbf{I} + \hat{\mathbf{Q}}_{xv} \mathbf{KC} + \mathbf{C}^T \mathbf{K}^T \hat{\mathbf{Q}}_{vx} & \mathbf{C}^T \mathbf{K}^T \\ \mathbf{KC} & -\hat{\mathbf{Q}}_{vv}^{-1} \end{pmatrix} \leqslant 0 \tag{2.58}$$

and $\theta > \theta^\star$:

$$\inf_{\mu \in \mathcal{S}} \sup_{0 \neq \underline{x} \in R^n} \left(\begin{pmatrix} \underline{x} \\ \mu(C\underline{x}) \end{pmatrix}^T Q \begin{pmatrix} \underline{x} \\ \mu(C\underline{x}) \end{pmatrix} \right) / (\|\underline{x}\|^2) = \theta^\star \tag{2.59}$$

Proof. From the cost function of Eq. 2.55 we can deduce the matrix Q_J for the minimax problem, so partitioned:

$$Q_J = \begin{bmatrix} Q_{xx} & Q_{xv} \\ Q_{vx} & Q_{vv} \end{bmatrix} \tag{2.60}$$

$$\begin{cases} Q_{xx} = \begin{bmatrix} \alpha_1+\beta_2 & -\beta_2 & ... & 0 & 0 \\ -\beta_2 & \alpha_2+\beta_2+\beta_3 & ... & 0 & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ... & \alpha_{M-1}+\beta_{M-1}+\beta_M & -\beta_{M-1} \\ 0 & 0 & ... & -\beta_{M-1} & \alpha_M+\beta_M \end{bmatrix} \\ Q_{vv} = diag(\gamma_1, \gamma_2, ...\gamma_M) \\ Q_{xv} = Q_{vx} = 0 \end{cases} \tag{2.61}$$

The matrices obtained refer to a dynamic system. To perform an offline computation of the trajectory, the system needs to be transformed from dynamic to static. Given the available data on the initial state and disturbance (i.e., $\underline{x}(0)$ and $\underline{w}(0)$, hereinafter denoted as x_0 and w_0 for notation simplicity), the following problem must be solved:

$$\inf_{v_0} \sup_{x_0,w_0} \left(\frac{x_0^T Q_{xx} x_0 + v_0^T Q_{vv} v_0 + x^T(1) Q_{xx} x(1)}{||x_0, w_0||^2} \right) \tag{2.62}$$

By writing $x(1)$ as a function of $x(0)$ from Eq. 2.49 and developing the computation, Eq. 2.62 may be rewritten as:

$$\inf_{v_0} \sup_{x_0,w_0} \left(\frac{x_0^T(Q_{xx} + A^T Q_{xx} A)x_0 + v_0^T(Q_{vv} + B^T Q_{vv} B) + x_0^T A^T Q_{xx} B v_0 + v_0^T B^T Q_{xx} A x_0}{||x(0), w(0)||^2} \right) \tag{2.63}$$

The numerator in Eq. 2.63 can be represented in the matrix form as:

$$\begin{bmatrix} x_0 \\ u_0 \end{bmatrix}^T \hat{Q} \begin{bmatrix} x_0 \\ u_0 \end{bmatrix} \tag{2.64}$$

More in detail, the matrix $\hat{Q}$ is composed as follows:

$$\hat{Q} = \begin{bmatrix} \hat{Q}_{xx} & \hat{Q}_{xv} \\ \hat{Q}_{vx} & \hat{Q}_{vv} \end{bmatrix} = \begin{bmatrix} Q_{xx} + A^T Q_{xx} A & A^T Q_{xx} B \\ B^T Q_{xx} A & R + B^T Q_{vv} B \end{bmatrix} \tag{2.65}$$

and it holds all the data needed to solve the LMI problem of Theorem 1.
Furthermore, Theorem 1 of [101] demonstrates that there exist linear decision $\mu_i(C_i x) = K_i C_i x$, for $i = 1, ..., N$ where the finite value θ^* of the game represented by equation (2.59) is achieved. □

Corollary 1.1. The control can be applied on a wider time horizon, i.e. $T > 1$, applying results in [102], Section VII.

The aforementioned robust control is applied to a five-vehicle platoon moving along a rectilinear path [103]. The whole system is willing to proceed at the cruising speed of $15m/s$, while initial speeds of individual vehicles are listed in Table 2.1.

Table 2.1: Vehicles' initial speed

Vehicle	speed [m/s]
1	18.79
2	17.55
3	17.67
4	15.59
5	14.66

The main goal of the control law is indeed to restore a safe inter-vehicular distance, which is assumed to be at least 29 meters according to Italian road traffic rules, that estimate the safe inter distance as:

$$d_{red}\,[m] \leq \frac{3 \times V\,[km/h]}{10} \qquad d_{green}\,[m] \geq \left(\frac{V\,[km/h]}{10}\right)^2 \qquad d_{red} < d_{yellow} < d_{green} \tag{2.66}$$

where d_{green} is the recommended distance, computed as the sum of the space traveled during the human reaction and the braking time, d_{yellow} is the reaction space only, and d_{red} is a critical inter-vehicular distance that can provoke collision in case of a sudden brake of the preceding vehicle. The unit mismatch arises because this equation is derived from a common practice in Italian road regulations, aimed at enabling drivers to quickly estimate the distance to keep with the preceding vehicle. A schematic representation is illustrated in Fig. 2.1.

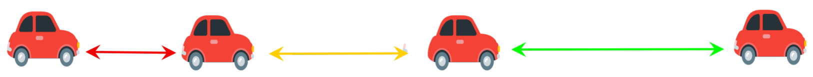
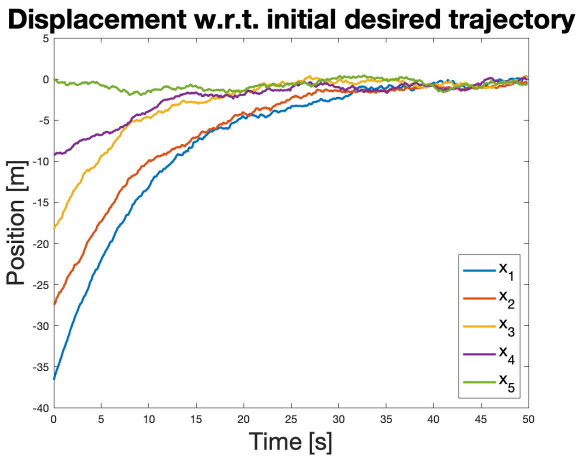


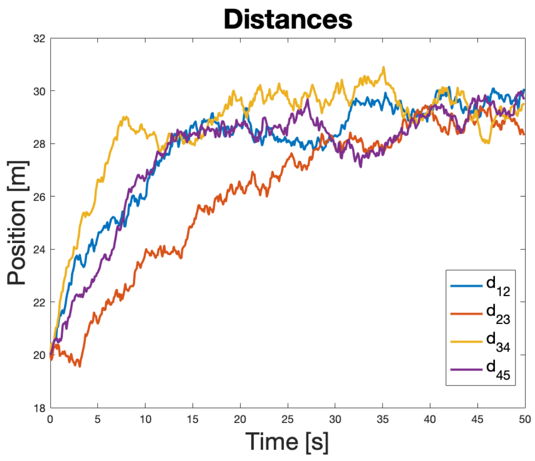
Figure 2.1: Safety zones division, according to the Italian road traffic rules

In other words, the robust control aims to space the vehicles with the recommended inter distance d_{green} with a smaller as possible speed variation.
Neighboring elements are suboptimally spaced, as there is a 20-meter gap between neighboring vehicles, caused by external factors prior to the simulation. Consequently, the platoon employs a robust LMI-based control to increase the spacing.
In this scenario, the noise is assumed to be additive white Gaussian, with zero mean and known covariance. Gains are held at unity for demonstration purposes, although in practical implementations they require empirical tuning based on their individual significance. It is advisable to give priority to restoring the optimal inter-distance when vehicles encounter

challenges exiting the yellow zone due to external factors. This adjustment will prompt them to reduce speed, improving stability while returning to the initial platoon formation. Fig. 2.2a illustrates the deviation of vehicle positions from their planned trajectories. Initially, there is a significant difference between desired and actual trajectories, but convergence is achieved over the course of the 50-second simulation. These results highlight the effectiveness of the robust controller in restoring the initial and optimal state of the vehicles in the string formation. Fig. 2.2b shows the inter-vehicle distances during the simulation. In this case, the predefined values (29 meters) are consistently maintained, aligning with the primary objective of the control law.

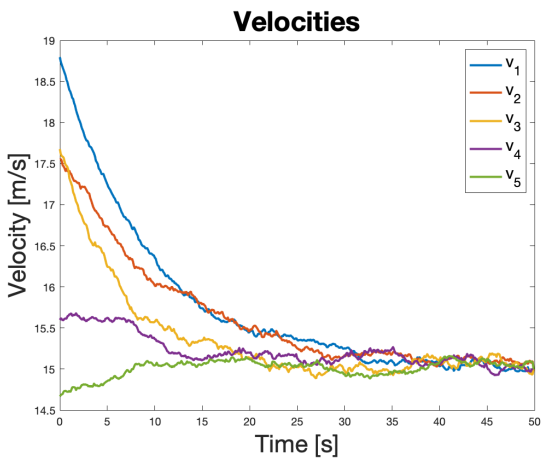


(a) Divergence in vehicles' position with respect to the planned one

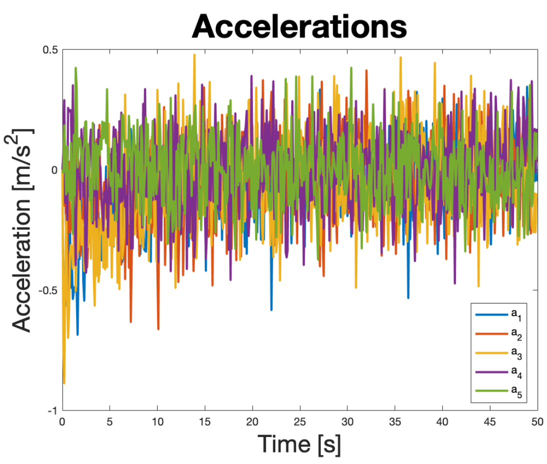


(b) Distance between neighboring vehicles

Figure 2.2: Platoon evolution over time



(a) Trend of the velocities



(b) Vehicles' accelerations

Figure 2.3: Trend of velocities and acceleration over time

Furthermore, Fig. 2.3a illustrates the velocity trends, aligning with expectations: the leading vehicles initially slow down to facilitate the correct repositioning of other platoon members, ultimately achieving stability. Subsequently, they resume cruising speed and, through other cooperative control techniques, they may increase both speed and inter-vehicle distance. Table 2.2 provides a summary of the velocities and distances between vehicles and their preceding platoon elements, validating the control algorithm. It is worth noting that the required acceleration, as depicted in Fig. 2.3b, remains within feasible limits (i.e., oscillating between -1 and 1), instilling confidence for testing the trajectory on a more realistic model.

Table 2.2: Vehicles' state at the end of the simulation

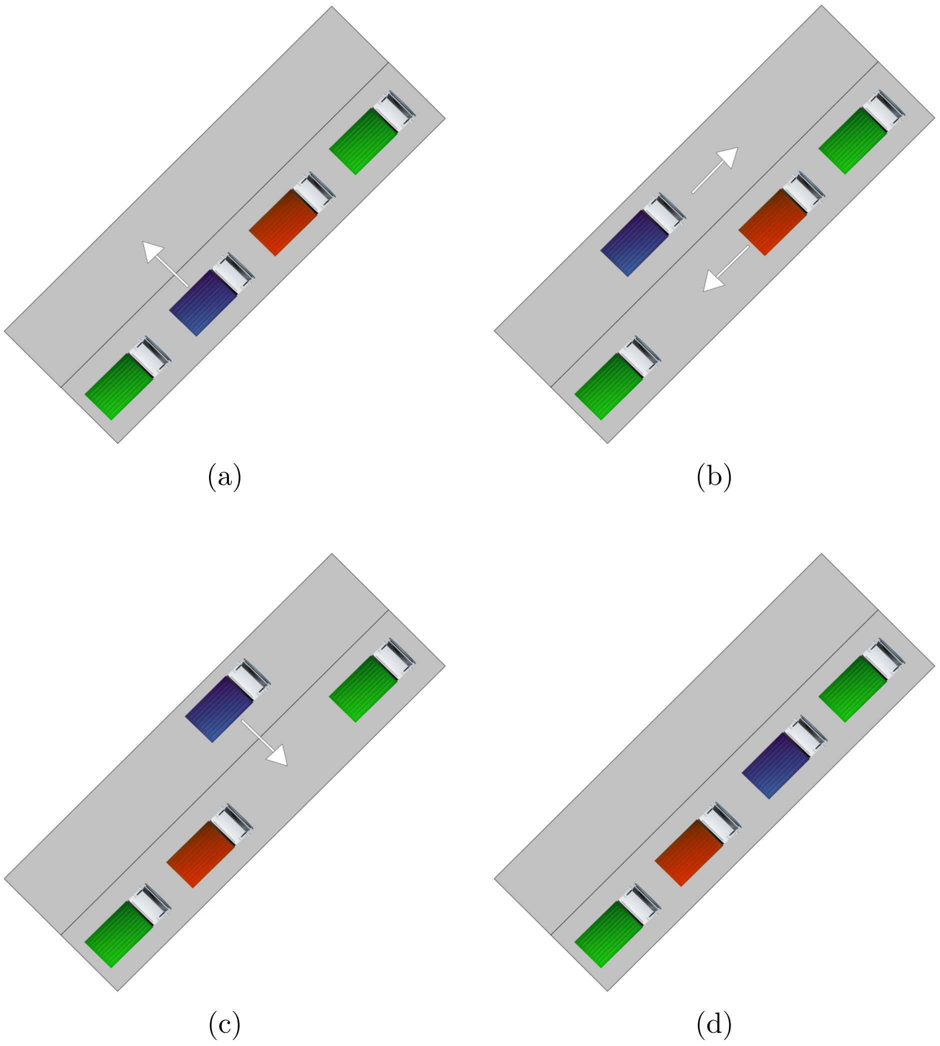
Vehicle	Velocity [m/s]	Distance from preceding car [m]
1	14.92	-
2	15.00	29.78
3	15.08	29.41
4	15.01	28.98
5	14.99	29.02

In conclusion, the resistance to disturbances and the linearity of the control law allows this algorithm to be used when fast modification of the planned trajectory is required due to unforeseen approaching among vehicles.

2.4 Comparison of LQT and PID techniques on highway scenario

After analyzing an initial one-dimensional scenario, the focus can be shifted to a more complex highway platoon maneuver. Specifically, an overtaking maneuver between two heavy-duty vehicles (HDVs) is considered [104]. It can be viewed as a space-constrained overtaking maneuver, as the vehicle wishing to overtake must:

- Position itself on the fast lane, exiting the string formation (Fig. 2.4b)
- Overtake the vehicle in front while the latter decelerates to favour the maneuver
- Settle at the correct distance between its neighboring vehicles (Fig. 2.4c)
- Return to the platoon lane (Fig. 2.4d)



(a) (b) (c) (d)

Figure 2.4: Overtaking maneuver with position constraints

The maneuver, graphically represented in Fig. 2.4 has to be performed keeping a similar speed with respect to the rest of the platoon and in a reasonable time frame. Moreover, for the whole time, vehicles involved in the swap have to prevent getting too close to their neighbors, thus endangering passengers' safety.
The role of each element can be summarized as follows:

- Vehicle #1 has to keep a constant speed and measure the distance to its follower, in order to improve the reconstruction of the surroundings
- Vehicle #2 decelerates in order to favor the overtaking of vehicle #3, while measuring

the distance from the leader and the last element of the platoon

- Vehicle #3 overtakes vehicle #2 and measures the distance from it and from the leader, in order to re-enter the string formation in the best position possible
- Vehicle #4, similarly to vehicle #1, has to keep a constant speed and increase the knowledge of the environment by providing its measurements

The Lomonossoff model discussed in Section 2.1.2 has been utilized to represent the nonlinear model of HDVs, and it is controlled using a PID controller of Section 2.2.3. The performance of this nonlinear model is then compared to that of the model described in Section 2.1.2.1, which employs a LQT controller of Section 2.2.2, benefiting from the linearity of the latter.

A four-vehicle platoon ($M = 4$) is considered, in which vehicles #2 and #3 are involved in a position swap. The first and last vehicles in the platoon maintain a constant speed of $v_{reg} = 22[m/s]$. This four-vehicle subset is a practical choice since it allows for the exchange of positions between two trucks while minimizing disruptions to adjacent platoon members. In larger platoons, the controller can focus on the four-vehicle subset only during the maneuver, keeping the other elements at a constant speed. The maneuver must prioritize the safety of the entire system, necessitating a minimum distance between adjacent vehicles, as computed in Eq. 2.66. According to this formulation, the minimum and the recommended distances are:

$$\begin{aligned} d_{min}[m] &= \frac{3 * v_{reg}[km/h]}{10} = 23.76m \\ d_{opt}[m] &= (\frac{v_{reg}[km/h]}{10})^2 = 62.73m \end{aligned} \tag{2.67}$$

Nevertheless, these bounds could potentially be reduced further, considering the quicker reaction times of unmanned vehicles compared to human-driven ones. While other rules exist for computing the optimal inter-vehicle distance, such as those outlined in Responsibility Sensitive Safety (RSS) principles widely discussed in the literature (e.g., in [105] and [106]), it is beneficial to begin with the recommended distances specified by traffic regulations, which currently serve as the minimum constraints on road safety. Naturally, in a scenario involving only unmanned vehicles, the possibility of shorter inter-vehicle distances can be explored.
The case study begins with an initial inter-vehicle spacing of $d = 30[m]$. This spacing is close to the critical bound (d_{min}) and represents a challenging scenario for executing an overtaking maneuver between platoon vehicles. During the first 5 seconds of the simulation, all vehicles maintain their regular speed, and at this point, vehicle #3 changes lanes to initiate the overtaking maneuver. The re-entry into the original lane is assumed to occur in the last 5 seconds of the simulation without altering the longitudinal displacement.

The performance of a PID controller on the continuous nonlinear system is presented and then compared to a control algorithm that deals with the linear discrete approximation of Lomonossoff's model and makes use of a LQT problem to compute the optimal control input.

2.4.1 Nonlinear system and PID control

The control of the nonlinear system is executed through individual PID controllers, each responsible for adjusting the speed of a vehicle from the nominal regime speed. These controllers convert the desired position into the corresponding tractive effort. Communication between vehicles is limited to sharing the current position, enabling the PID controllers to formulate appropriate control actions to achieve the desired position. Specifically, the desired position (x_d) is defined relative to the actual position and the position of the platoon leader, or the position of the last platoon element, along with the inter-vehicle distance, which is computed based on the nominal regime speed. At each sampling instant k, it can be expressed as follows:

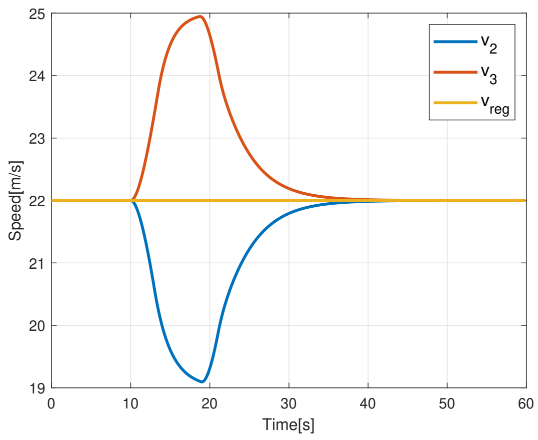
$$\begin{cases} x_2^d(k) = x_1(k) - 2d \\ x_3^d(k) = x_4(k) + 2d = x_1(k) - d \end{cases} \tag{2.68}$$

Reference trajectory of vehicle #3 should be written with respect to the position of the leader, since unexpected behavior can arise when dealing with a large-scale platooning, as demonstrated in [41]. However, considering the small number of vehicles involved in the platoon, both formulations give the same results.

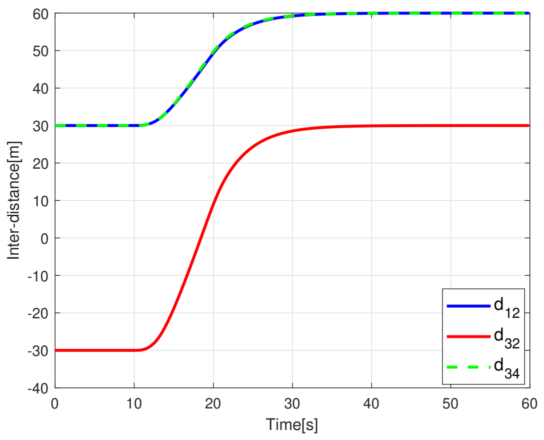
Table 2.3: PID coefficients

Proportional	2.754
Integral	0.484
Derivative	2.986
Filter coefficient	9.300

The parameters of the continuous-time PID controller were tuned using the Matlab/Simulink tool by linearizing the plant near the equilibrium point corresponding to the nominal regime speed. However, it is important to note that unexpected behavior may occur at significantly different speeds from the initial one. The simulation employs the values listed in Table 2.3. A filter coefficient is applied to enhance the performance of the derivative term, which is not implemented as a pure derivative due to its sensitivity to noise. Additionally, a rate limiter is introduced to prevent abrupt changes between consecutive sampling instants. Output saturation is used to keep the vehicle near its equilibrium point and enhance the overall realism of the control input in the virtual environment.



(a) Vehicles' speed: nonlinear system



(b) Inter-vehicle distances: nonlinear system

Figure 2.5: Platoon evolution with nonlinear model and PID controller

Fig. 2.5a shows the trend of the velocities. As expected, there is a slight acceleration from the third vehicle, while the other starts decelerating to favor the overall maneuver. The behavior is almost specular as both vehicles have the same PID gains and their desired positions are symmetric with respect to the center of the platoon.
Fig. 2.5b confirms the effectiveness of the nonlinear controller, as the distance between vehicles varies according to the expectation and it settles on multiples of 30 meters (based on the pair of vehicles analyzed), ensuring that the initial inter-vehicle distance is maintained.
The objective is achieved smoothly and quickly, without abrupt changes in acceleration thus preserving passenger comfort.

2.4.2 Linear system and LQT control

The key distinction of the linear controller compared to the previous one is its reliance on time-varying system information, operating on a discretized model detailed in Section 2.1.2.1. This model serves as an approximation of the actual system. Specifically, the vehicle's state is utilized to linearize the system around the operating point $v(\bar{t}_p)$ and provide input to the Linear Quadratic Tracking (LQT) problem. Consequently, it is assumed that the velocity trends for the nonlinear evolution are known and used as a reference signal for the tracking algorithm.

The knowledge of velocity trends is a strong assumption, but it is reasonable as the behavior depends on the actual speed of the vehicles. Thus, it is possible to have a set of different maneuvers based on the initial regime speed of the other vehicles of the platoon.
The LQT necessitates a cost function to assign priorities to tracking specific state variables and controlling costs. For each sampling instant the cost function is formulated as follows:

$$\begin{aligned} J = \sum_{i=1}^{M}\sum_{k=1}^{K}(\alpha_i(x_i(k) - x_i^d(k))^2 + \beta_i(v_i(k) - v_{reg})^2 + \gamma_i f_i(k)) + \\ \sum_{i=1}^{M}(\alpha_i(x_i(K+1) - x_i^d(K+1))^2 + \beta_i(v_i(K+1) - v_{reg})^2) \end{aligned} \quad (2.69)$$

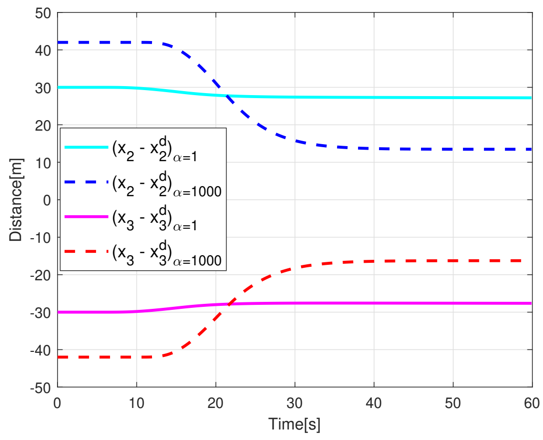
where α, β, and γ are gain values that are tuned to prioritize specific elements of the summation, and K represents the control horizon for the LQT problem. A higher value of K results in smoother system response but reduces its responsiveness. In the case study, K is set to 10 to manage computational costs efficiently.
The desired positions, denoted as x_i^d, are determined following Eq. 2.68 for vehicles involved in the swap maneuver. For the outer vehicles, x_i^d is computed as a constant displacement between consecutive sampling instants. The regime speed, denoted as v_{reg}, is fixed at $22m/s$ for all platoon members in the case study. Specifically, the first term is the quadratic deviation from the desired trajectory, which for inner vehicles x_i^d is computed as in Eq. 2.68 and for outer vehicles is not considered (i.e. $\alpha_1, \alpha_4 = 0$), the second term is the deviation from the desired speed and their trend is supposed to be known, and the last term regards the minimization of the input which translates in the minimum tractive effort to be applied to accomplished the goals. The second summation is needed to represent the final control instant for the state.
The equivalent of the cost function in matrix form is:

$$Q = \begin{bmatrix} \alpha_3+\alpha_2 & 0 & -\alpha_2 & 0 & -\alpha_3 & 0 & 0 & 0 & -\alpha_3 d - 2\alpha_2 d \\ 0 & \beta_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\alpha_2 & 0 & \alpha_2 & 0 & 0 & 0 & 0 & 0 & 2\alpha_2 d \\ 0 & 0 & 0 & \beta_2 & 0 & 0 & 0 & 0 & -\beta_2 v_{reg} \\ -\alpha_3 & 0 & 0 & 0 & \alpha_3 & 0 & 0 & 0 & \alpha_3 d \\ 0 & 0 & 0 & 0 & 0 & \beta_3 & 0 & 0 & -\beta_3 v_{reg} \\ 0 & 0 & 0 & 0 & 0 & 0 & \alpha_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta_4 & 0 \\ -\alpha_3 d - 2\alpha_2 d & 0 & 2\alpha_2 d & -\beta_2 v_{reg} & -\alpha_3 d & -\beta_3 v_{reg} & 0 & 0 & 0 \end{bmatrix}$$

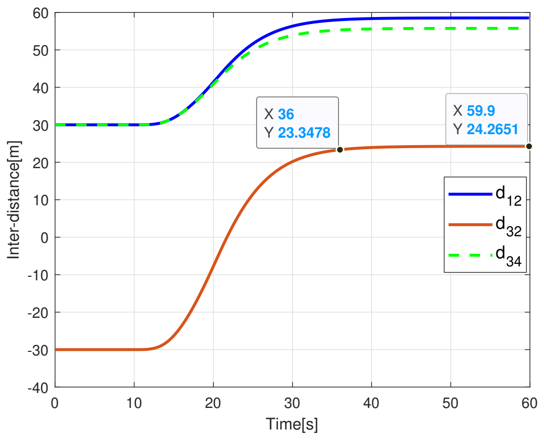
$$R = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.70)$$

Note that $Q \in \mathbb{R}^{2M+1}$ and $R \in \mathbb{R}^{M+1}$ due to nonquadratic terms present in the cost function that require to increase the size of the matrix and the state vector, following the procedure in [107].

The control technique is inherently centralized, with the system state composed of all platoon vehicles, including those traveling at constant speeds. This centralization ensures optimal platoon behavior from a collective perspective, prioritizing overall system safety over individual elements. In the case study, a strong emphasis has been placed on tracking speeds over controlling costs by setting high β gains. It should be noted that setting all α gains to zero transforms the platoon control problem into M individual vehicle control problems, as no constraints related to inter-vehicle distance exist.



(a) Difference between actual and desired position with $\alpha = 1$ and $\alpha = 1000$



(b) Inter-vehicle distances: linear controller

Figure 2.6: Platoon evolution with nonlinear model and PID controller

With unitary gains, the control technique cannot execute the vehicle swap, as illustrated in Fig.2.6a. This occurs because the control algorithm finds the maneuver costly and thus not favorable. On the other hand, when greater importance is placed on the maneuver (i.e., when gains for tracking velocity trends significantly outweigh control cost gains), the inter-vehicle distance evolves as shown in Fig.2.6b, with markers highlighting behavior at the midpoint and the end of the simulation.

The overall objective is achieved with minor reductions in inter-vehicle distances. This does not compromise the system's safety, as each element maintains a distance greater than d_{min} from the following vehicle. This is particularly valuable given the high regime speed and the initially unfavorable conditions that may not suggest such a maneuver. Increasing the initial spacing between vehicles yields safer results, as it provides more room for the exchange.

2.5 Platoon integration in macroscopic environment

In the upcoming section, the focus will shift from the analysis of platooning maneuvers to the integration of platooning into the control of highway traffic within the macroscopic road environment described in Section 2.1.3. This transition expands the scope of the research, offering valuable insights into the potential advantages and challenges associated with platoon-based traffic management on a larger scale. From the platooning control purpose, this model serves as a bridge between the micro-level interactions within individual nodes, namely the platoon, and the macro-level behavior of the entire network, the traffic flow. The objective is to develop a hierarchical control scheme to improve the travel performance of truck platoons, in terms of travel times, comfort, and safety [108]. As depicted in Fig. 2.7, the control of each platoon is implemented by means of a two-level control architecture located in the leader vehicle of each platoon. More in detail, the high level of control, that is a PI controller of Section 2.2.3, defines the reference speed of the platoon based on the traffic conditions detected in a portion of the freeway downstream of the platoon itself. Based on this speed reference profile, the low control level, that is a LQT controller of Section 2.2.2, defines the accelerations that each vehicle in the platoon must actuate to achieve the desired speed while maintaining the inter-vehicular distances ensuring the necessary platoon safety conditions. Indeed, at the low control level, the optimal control is achieved with respect to the global behavior of the platoon, thus forbidding selfish actions performed by individual vehicles.

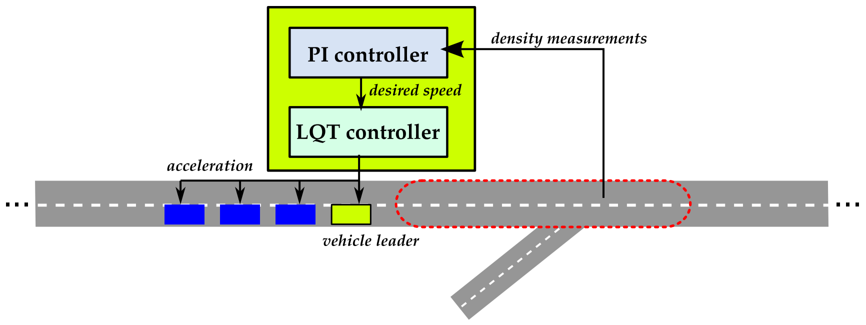


Figure 2.7: Sketch of the proposed control scheme.

It must be noted that combining the high control level defined on the basis of traffic conditions with the low control level accounting for the conditions of each single vehicle allows for regulation of the vehicles in the platoon in order to guarantee safety and, at the same time, to improve the travel performance of the platoon. Indeed, the reference speed defined at the high control level considering the platoon in an aggregate way may be unfeasible in practice, where a platoon is instead composed of several vehicles with different dynamic characteristics. The combination of the two control levels allows to overcome this

criticality and to find the optimal behavior of the vehicles according to the reference speed and their dynamic behavior.
The goal of the proposed control scheme is to improve the driving experience of individual platoons by trying to reduce the time spent in congested area and by limiting abrupt speed variations. These aspects are crucial for improving safety, reducing traffic emissions, and platoon fuel consumption due to driving in congestion.

2.5.1 High-level controller: PI-type platoon speed feedback

The objective of the high control level is to define the speed for each platoon to improve its performance indices. This speed is defined by means of a PI-type feedback controller based on traffic conditions measured instantaneously and continuously downstream of each platoon. Specifically, at each time step k the vehicle leader of platoon z receives measurements of the traffic densities detected over a number of sections $\mathcal{N}^z$ that are immediately downstream of the platoon position. Based on these measurements, it computes the average density $\varrho^z(k)$ over the subset of sections $\mathcal{N}^z$, which define the reference speed $\hat{v}^z(k)$ to transmit to the low-level controller. To this end, the subset of sections used to compute the average density is defined as $I^z(k)$. This subset starts at $i^z(k)$, i.e., the section where platoon z is located at time step k, and lasts for $\mathcal{N}^z$ sections. The average density $\varrho^z(k)$ is calculated as

$$\varrho^z(k) = \frac{\sum_{i \in I^z(k)} \bar{\rho}_{m,i}(k)}{\mathcal{N}^z} \tag{2.71}$$

Hence, the reference speed of platoon $\hat{v}^z(k)$ at time step k is given by

$$\hat{v}^z(k) = v^z(k-1) + K_P \left[\varrho^z(k-1) - \varrho^z(k)\right] + K_I \cdot \left[\hat{\rho} - \varrho^z(k)\right] \tag{2.72}$$

where K_P and K_I are the gain parameters of the controller, $v^z(k-1)$ is the average speed actuated by platoon z in the previous time step and where the density set-point $\hat{\rho}$ is generally set equal to the critical density ρ^{cr}. Before being transmitted to the low-level controller, the speed defined with Eq. 2.72 must be compliant with the bounds referring to the minimum values of speed allowed for platoon z and the current speed of the section in which the platoon is located at time step k or the maximum platoon speed, i.e.

$$v^{z,\min} \leq v^z(k) \leq \min\left\{v^{z,\max}, v_{i^z(k)}(k)\right\} \tag{2.73}$$

2.5.2 Low-level controller: Linear Quadratic Tracking

Analyzing the traffic flow from the microscopic point of view, platoons represent the main bottleneck of the system. This happens due to the higher number of constraints that involve many vehicles at a time. Therefore, platoons require specific control techniques

because the optimality of the control regards the whole set rather than the individual truck and their performance greatly influences traffic flow.
The discretized model of vehicles belonging to platoons is the second-order kinematic model of Section 2.1.1 and their state equations, referring to a single vehicle j, can be translated into matrix form:

$$A_j^z = \begin{bmatrix} 1 & \overline{T} \\ 0 & 1 \end{bmatrix} \quad B_j^z = \begin{bmatrix} \frac{1}{2}\overline{T}^2 \\ \overline{T} \end{bmatrix} \tag{2.74}$$

with $\overline{T}$ the sampling time referred to the dynamics of the individual vehicles present in the platoon. It has to be noted that $\overline{T}$ must be consistently smaller than T, the METANET sampling time, to allow vehicles to react promptly to some unexpected situations during their trip.
Consequently, the system matrices gathering the dynamics of all the vehicles composing platoon z are represented as

$$\underline{x}^z(h+1) = A^z\underline{x}^z(h) + \overline{T}B^z\underline{u}^z(h) \tag{2.75}$$

with A^z and B^z being the diagonal concatenation of A_j^z and B_j^z, respectively. Moreover, each vehicle can access information on its own state and on its neighbors, as in the case study of Section 2.4, and also about the leader.
The speed provided by the PI controller is the signal to be tracked from the low-level controller, which handles the platoon displacement by exploiting a centralized LQT. The aim of the control algorithm is to reach the target speed in the shortest possible time while keeping an adequate inter-vehicle distance between elements.
For each platoon z an optimization problem is solved by minimizing the following objective function:

$$J = \sum_{j=1}^{n^z}\sum_{h=1}^{H}\left[\alpha^z|\underline{v}_j^z(h) - \underline{w}_j^z(h)|^2 + \omega^z\underline{u}_j^z(h)^2\right] + \sum_{j=1}^{n^z-1}\beta_1^z|\underline{p}_j^z(h) - \underline{p}_{j+1}^z(h) - \delta_j|^2 + \sum_{j=3}^{n^z}\beta_2^z|\underline{p}_1^z(h) - \underline{p}_j^z(h) - (j-1)\delta_j|^2 \tag{2.76}$$

The first term aims to track the desired state. Specifically, for each vehicle j composing platoon z, $\underline{x}_j^z$ is the state vector composed of position and speed, while $\underline{w}_j^z$ is the reference signal composed again of position and speed. The reference value of the speed $\hat{v}^z(k)$ communicated by the high control level is kept constant for a number of time steps equal to $T/\overline{T}$, so the reference position is retrieved from the transmitted value of $\hat{v}^z(k)$. The second term is defined to limit abrupt changes in the control trajectory, while the third term is included in the objective function to implement the inter-distance constraints. The parameter δ_j represents the desired inter-vehicle distance for vehicle j and it is calculated using the two-second rule. This rule computes the distance to maintain as the current

speed of vehicle j multiplied by two seconds, relative to the desired speed of the vehicle. It is worth noting that, given the faster reaction times of autonomous vehicles compared to human drivers, this inter-vehicle distance could potentially be reduced. The last term enforces position constraints by adding terms related to the inter-vehicle distance that each element has to preserve with respect to the leader.
Finally, the parameters α^z, β_1^z, β_2^z, ω^z are gains to weigh the different cost function terms. The cost function can be put in matrix form to obtain the matrices Q^z and R^z required to implement the LQT problem that supplies the optimal acceleration, saturated with feasible values, to prompt to each vehicle.

$$J = \underline{x}^{z\intercal} Q^z \underline{x}^z + \underline{u}^{z\intercal} R^z \underline{u}^z \tag{2.77}$$

2.5.3 Traffic performance controlling platoons speed

As introduced before, the goal of this control architecture is to improve the operation of truck platoons, in terms of time spent in the congested area and in terms of speed variations. In the following, two suitable performance indices are introduced to quantify the performance of the controller, defined on the basis of the Micro-Macro METANET model.
The time spent in the congested area by platoon z is denoted as τ^z and given by

$$\tau^z = T \sum_{k=0}^{K} \eta^z(k) \tag{2.78}$$

where $\eta^z(k)$ is equal to 1 if platoon z is, at time step k, in a section in which the traffic density exceeds the critical value, i.e. $\bar{\rho}_{m^z(k),i^z(k)}(k) \geq \rho^{\text{cr}}$.
The second performance index computes the smoothness of the speed profile of platoon z, which is strongly related to comfort levels. This index is denoted as σ^z and is given by

$$\sigma^z = \sum_{k=1}^{K} \left(v^z(k) - v^z(k-1)\right)^2 \tag{2.79}$$

The effectiveness of the platoon speed controller can be computed considering the entity of the reduction of τ^z and σ^z compared with the uncontrolled case.
The freeway stretch adopted to test the control framework, which is depicted in Fig. 2.8, is 20 [km] long and is divided into $N = 40$ sections each of which has a length of 0.5 [km]. The stretch has three on-ramps, located respectively at kilometers 11, 13 and 15, and an exit ramp at kilometer 14. The stretch under consideration has three lanes except for one kilometer where only two lanes are present. Specifically, the narrowing is located from kilometer 11.5 to kilometer 12.5. The presence of four platoons is considered. An

equivalent occupancy $o^z = 10$ [veh/km/lane], which corresponds to $n^z = 5$ vehicles, has been considered $\forall z$. The arrival of the platoons at the freeway is scheduled at minutes 75, 97, 107 and 117 respectively.

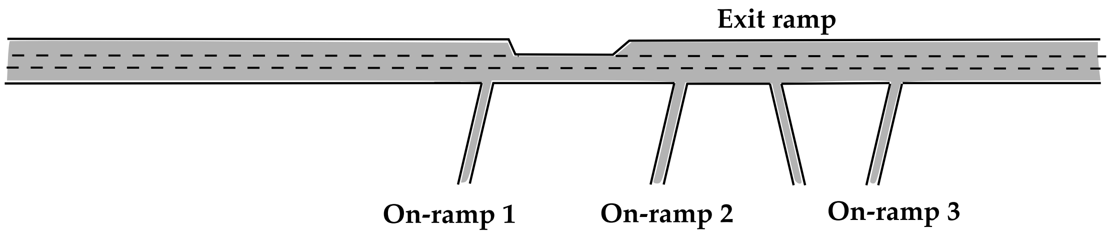
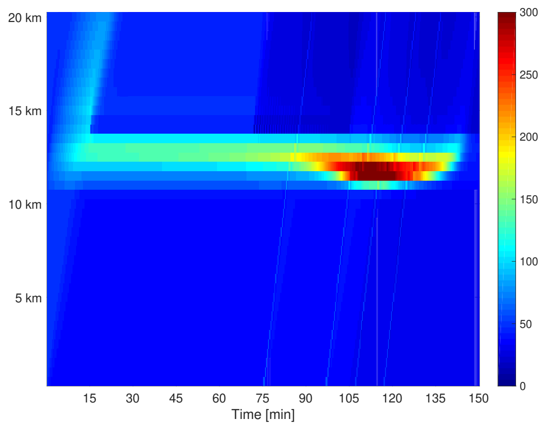
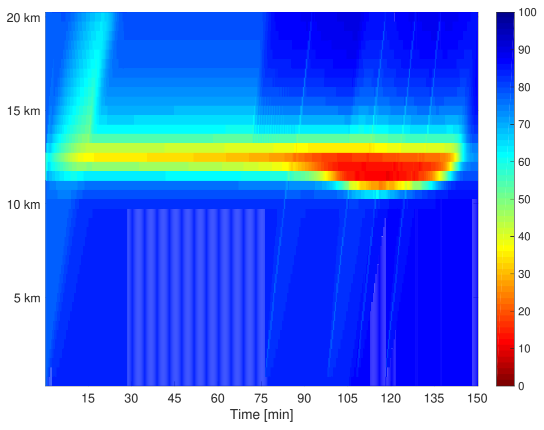


Figure 2.8: Sketch of the freeway network adopted for the case study.

The simulation has been conducted over a time horizon of two and a half hours. The sample time T adopted for the traffic simulation model and to compute the reference value of speed of each platoon has been set equal to 10 [s] ($K = 900$ time steps). Moreover, the following values of the traffic model parameters have been adopted: $v^{z,\max} = 85$ [km/h], $\forall z$, $v_i^{\mathrm{f}} = 100$ [km/h], $\rho_i^{\max} = 200$ [veh/km/lane], $\rho_i^{\mathrm{cr}} = 50$ [veh/km/lane], $\forall i$, $q_0^{\max} = 6000$ [veh/h], and $r_i^{\max} = 1800$ [veh/h] for all on-ramps.
By using the simulation model, it is possible to reproduce the traffic behavior in the stretch. As it is possible to observe in Fig. 2.9, which shows the time and space evolution of traffic density and speed, the platoons encounter a state of congestion mainly due to the lane drop at kilometer 11.5.



(a) Traffic density in the freeway stretch



(b) Traffic speed in the freeway stretch

Figure 2.9: Traffic indicators of congestion in the freeway stretch

As for the microscopic representation of platoons, there is a need for a shorter sampling time ($\overline{T} = 50$ [ms]) to ensure the responsiveness of vehicles to unexpected events. This allows the platoons to evolve coherently between two consecutive Micro-Macro METANET time intervals and allows the low-level controller to work with a reasonable control horizon, $H = 5$, in order to pursue the desired speed.
The tuning of α^z, β_1^z, β_2^z and ω^z can affect the platoon behavior in response to the evolution of the traffic flow. In this case study, the gains have been considered equal for all the platoons in the simulation. Primary importance has been given to tracking the reference speed ($\alpha^z = 5$) and to the minimization of the input $\omega^z = 5$. The values $\beta_1^z = 1$ and $\beta_2^z = 1$ have been used to force a different behavior between elements of the platoon by giving secondary importance to the optimal inter-vehicle distance. Note that, by posing β_1^z and β_2^z equal to zero each vehicle can be considered independent by the other elements.
At the end of a Micro-Macro METANET sampling time interval, the average speed of the leader within 10 seconds is provided to the Micro-Macro METANET, and the new state of the platoon, considering the evolution of the traffic, is retrieved to start again the LQT control algorithm.

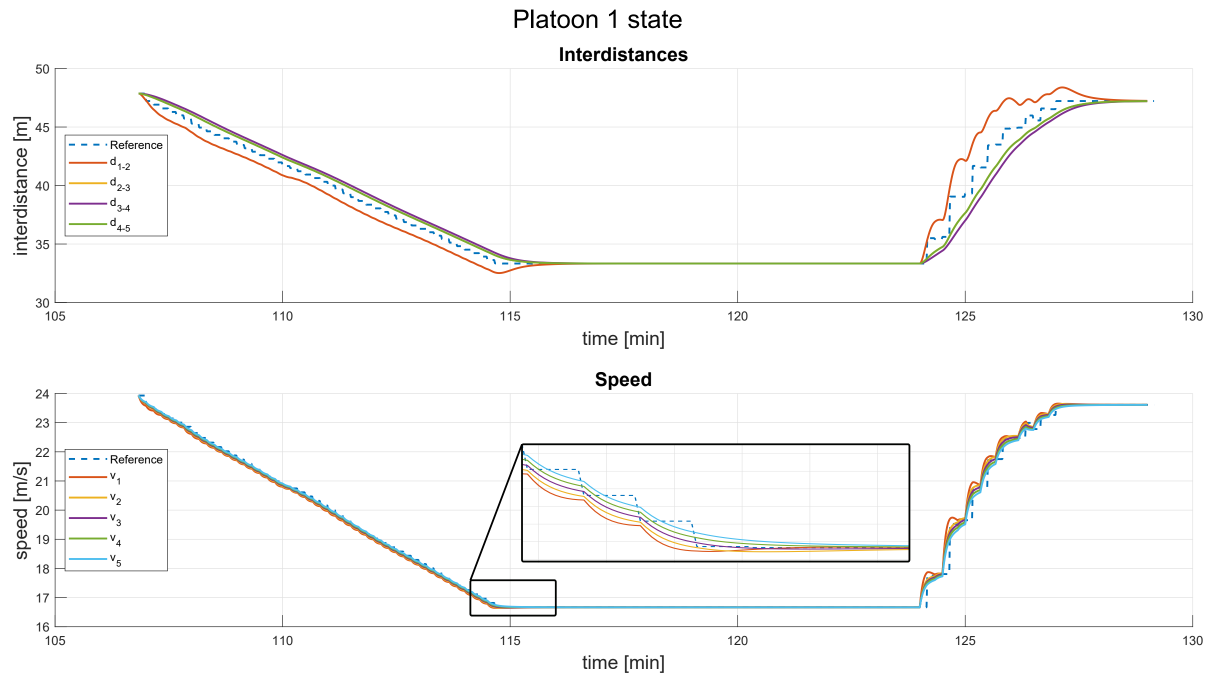


Figure 2.10: Evolution over time platoon #1

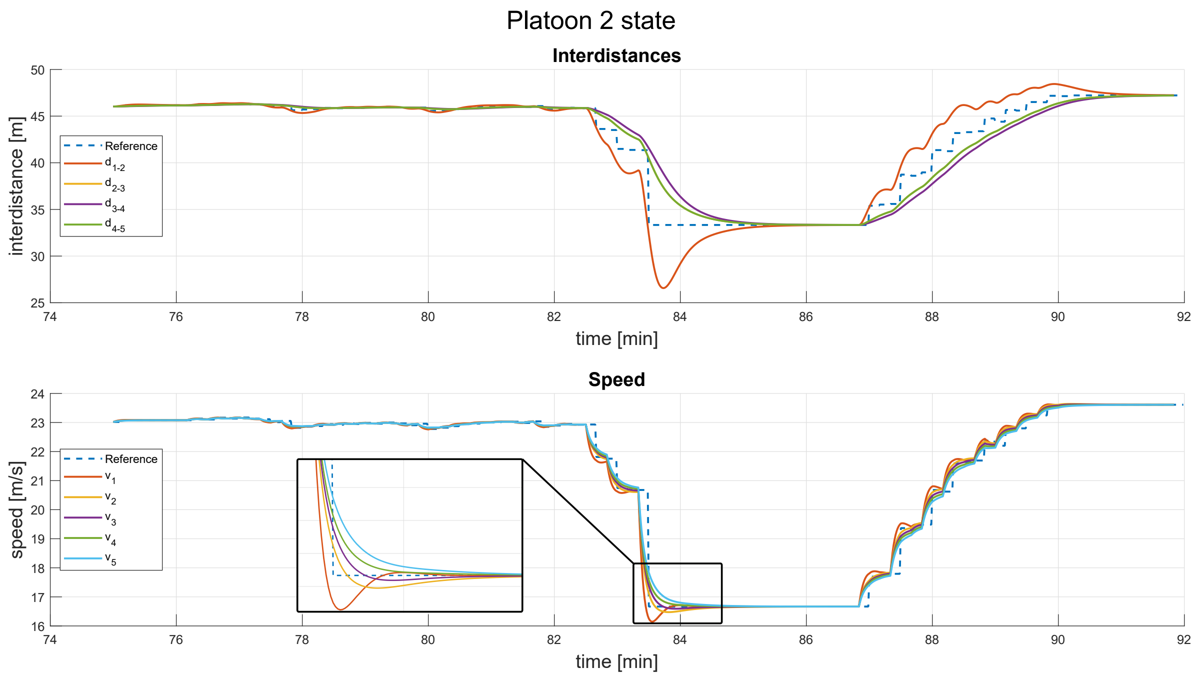


Figure 2.11: Evolution over time platoon #2

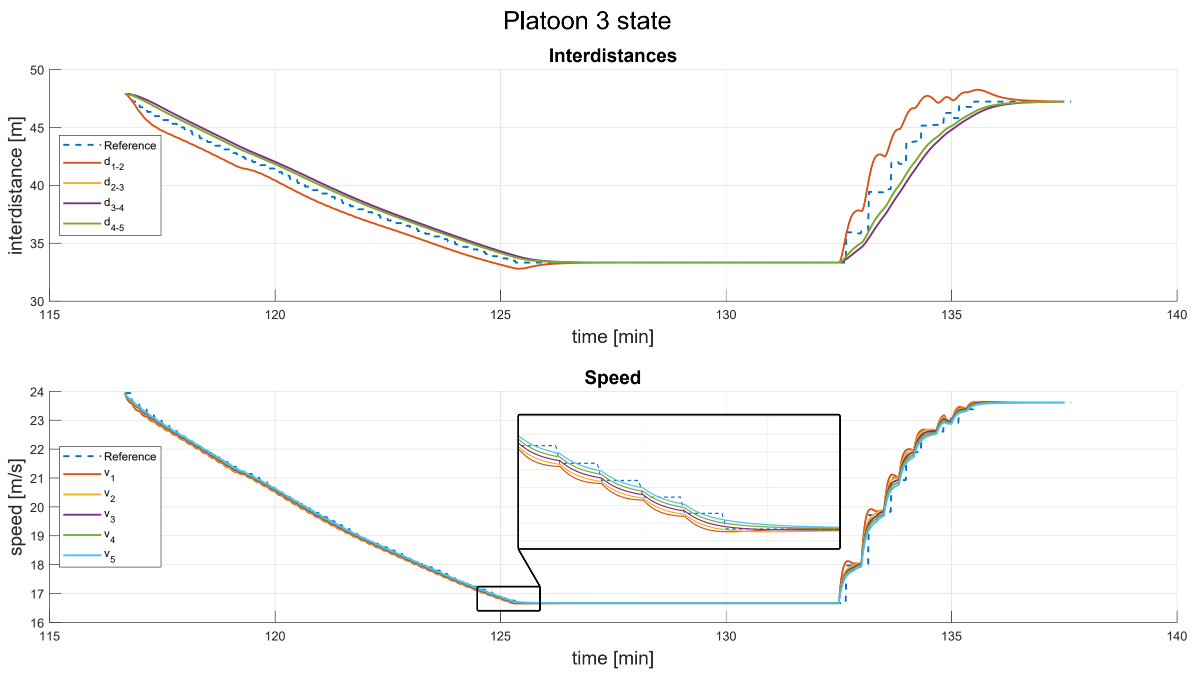


Figure 2.12: Evolution over time platoon #3

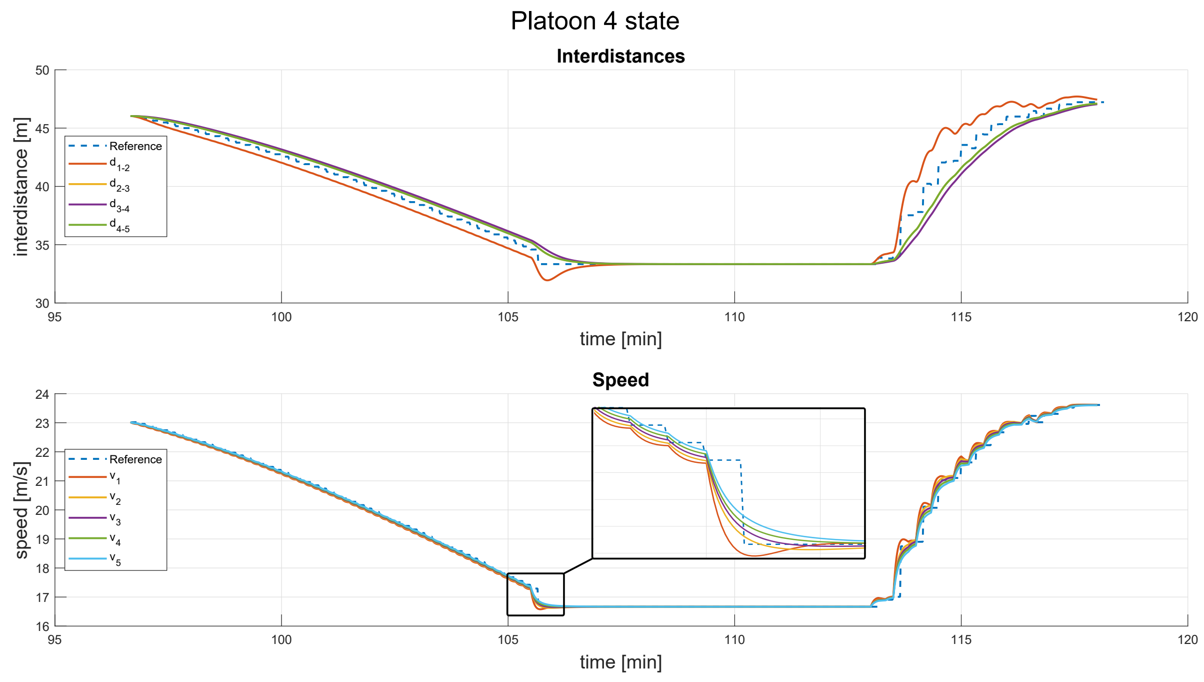


Figure 2.13: Evolution over time platoon #4

Figs. 2.10-2.13 shows the evolution of each platoon over time, with respect to their entrance and exit time in the main flow. The optimal inter-vehicle distances are broadly maintained throughout the whole time of simulation without endangering passengers safety. In other words, the inter-distances do not become too low with respect to the optimal one. Moreover, the desired speed is well-followed even in case of abrupt changes. The zoom in the speed profile denotes the slight differences among elements of the platoon. This suits the expectation of having a set of vehicles moving at a similar speed.

Table 2.4: Performance parameters

Platoon	τ^z no-control	τ^z control	% of improvement
$z = 1$	6.33	6.00	5.3
$z = 2$	1.83	1.83	0.0
$z = 3$	5.33	4.83	9.4
$z = 4$	5.83	5.50	5.7

Platoon	σ^z no-control	σ^z control	% of improvement
$z = 1$	17584	16892	3.9
$z = 2$	15548	15530	0.1
$z = 3$	17659	16741	5.2
$z = 4$	16702	16158	3.3

Finally, the previously defined performance indexes are improved in three out of four platoons ($z = 1, 3, 4$), as shown in Table 2.4. This depends on the traffic conditions encountered by the platoons, but in no case does it lead to a worsening in performance.

2.6 Application of event-based controller on WMR in driving routines

A further step in vehicle modeling involves transitioning from theoretical models to practical implementations. WMRs are valuable tools for this purpose, serving as a practical platform to validate and refine autonomous vehicle concepts in real-world scenarios. The models discussed in the previous sections can be validated through testing on simplified physical hardware. This initial testing provides an initial assessment of the effectiveness of the control techniques proposed in the following section.
WMRs offer a valuable platform for simulating the behavior of autonomous vehicles in embryonic-stage projects. These projects typically involve the early development and testing of algorithms, sensors, and control systems before the physical implementation of a full-scale autonomous vehicle. They provide an accessible and cost-effective means to replicate many of the challenges and scenarios encountered by autonomous vehicles in real-world environments. Indeed, developing autonomous vehicles can be prohibitively expensive, especially in the early stages. WMRs are generally more affordable, making them an ideal choice for prototyping and experimentation without incurring the high costs associated with full-scale vehicles. Moreover, they are highly modular and easily customizable.

Researchers and developers can quickly iterate on hardware and software components, allowing for rapid prototyping of different autonomous vehicle configurations while replicating real-world scenarios at a sufficiently realistic level. They may even be equipped with a wide array of sensors, including lidar, cameras, and GPS, similar to those used in autonomous vehicles to favor data fusion, and in turn, their data can be valuable for training machine learning models and validating autonomous systems. Lastly, WMRs play a huge role in teaching and training, helping students familiarize themselves with hands-on experience in developing and testing autonomous algorithms.

In the final section of this chapter, event-based algorithm techniques of Section 2.2.4 are applied to a physical hardware platform, representing a simplified model of an autonomous vehicle. Importantly, the analysis extends beyond simulated environments, as real-world scenarios are tested [103]. This transition enhances the value of the previous work, as it opens the possibility to assess control techniques on simpler yet more cost-effective physical hardware, providing an initial analysis of algorithm reliability.

The Freenove 4WD mobile robot of Fig. 2.14 is used to represent the vehicle's dynamics. It is a four-wheeled non-steering mini-car, 21 cm long and 15 cm wide with a mass of 400 grams, equipped with an ultrasonic sensor, wheel encoders, and line sensors.

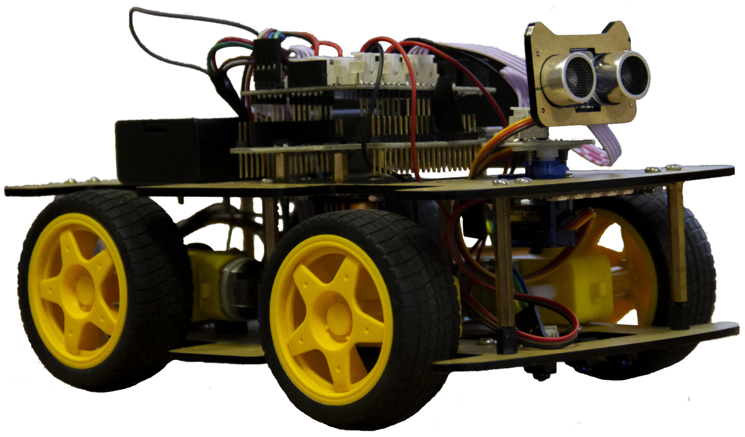


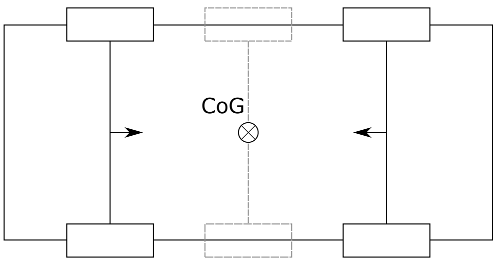
Figure 2.14: Wheeled Mobile Robot used for testing

Each wheel is equipped with its own DC motor, but the controller simultaneously provides the same power to the wheels on the same side. This design allows the mini-car to rotate in place, changing its direction effectively. Because of the hardware architecture, wheels

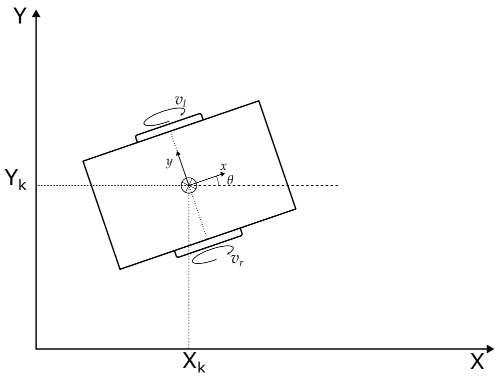
on the same side consistently receive identical power, resulting in uniform angular speed. Consequently, we can represent the wheels using a simplified model with only one wheel per side, conveniently positioned near the Center of Gravity (CoG), as illustrated in Fig. 2.15a.
Under these assumptions, the dynamics of the WMR, graphically represented in Fig. 2.15b, can be described by:

$$\begin{cases} \dot{x} = \frac{R}{2}(v_r + v_l)\cos\theta \\ \dot{y} = \frac{R}{2}(v_r + v_l)\sin\theta \\ \dot{\theta} = \frac{R}{L}(v_r - v_l) \end{cases} \tag{2.80}$$

where x and y are the longitudinal and lateral position of the robot with respect to the CoG, θ denotes its heading, R is the wheel radius, L is the wheelbase and v_l,v_r are the left and right angular wheel velocities.



(a) Schematic representation of the mini-car. Wheels on the same side have always the same rotation speed, thus one wheel per side located next to the CoG is supposed.



(b) Scheme representing the mechanical aspects involved in the model defined by Eq. 2.80

Figure 2.15: Representations of the WMR and its mechanical characteristics

More in detail, the model is the differential version of the kinematic model of Section 2.1.1. The WMR, albeit in a limited sense, serves as a representation of an autonomous vehicle. On the other hand, it can also serve as a valuable model for agents responsible for transporting products within a manufacturing system, as further explored in Chapter 4. The WMR is equipped with a Nucleo board microcontroller with a frequency of 32 kHz and various sensors:

- **Ultrasonic sensor**: The HC-SR04 Ultrasonic Ranging Module integrates both an ultrasonic transmitter and a receiver. The transmitter serves to convert electrical signals into high-frequency sound waves, while the receiver performs the opposite function. The module operates on the principle that ultrasonic waves reflect upon encountering obstacles. Distance measurement is achieved by calculating the time interval between the transmission and reception of ultrasonic waves after they encounter an obstacle. This time interval, denoted as 't,' represents the complete duration of the ultrasonic wave's journey. Given that the speed of sound in air is a constant ($v = 343m/s$), distance 'd' between the Ultrasonic Ranging Module and the obstacle can be calculated using the formula: $d = vt/2$.

- **Line-tracking sensor** The line-tracking sensor comprises three reflective optical photodiodes that emit and receive infrared light, which reflects off a surface and detects color based on light intensity. Each optical sensor produces a high-level signal upon detecting a black surface and a low-level signal when the surface is white. A lookup table is used to convert a three-bit value into a controller value (Table 2.5). Negative values indicate a control action towards the left, while positive values indicate a control action towards the right, weighted with different control intensities. This helps keep the robot on the lane when it deviates from the centerline.

left	center	right	control intensity
0	0	0	0
0	0	1	2
0	1	0	0
0	1	1	1
1	0	0	-2
1	0	1	0
1	1	0	-1
1	1	1	0

Table 2.5: Look-up table used for the line-tracking sensor

- **Wheels encoders**: The encoders used are rotary incremental with an optical interrupter that detects the rotation of a wheel with holes. Each time the encoder detects a hole in the wheel, it sends an interrupt. By counting the number of these interruptions (ticks) during a specific time interval, the traveled distance, speed, and wheel rotation speed can be retrieved. Denoting the total number of ticks at time k as $\mathbb{t}(k)$, the rate is:

$$\Delta\mathbb{t}(k) = \mathbb{t}(k) - \mathbb{t}(k-1) \tag{2.81}$$

At each instant, the traveled distance $\mathbb{d}$, speed $\mathbb{v}$, and rotational speed ω are:

$$\mathbb{d}(k) = \frac{2\pi R}{N_{\mathbb{t}}}\mathbb{t}(k) \tag{2.82}$$

$$\mathbb{v}(k) = \frac{\Delta\mathbb{t}(k)}{Ts} \tag{2.83}$$

$$\omega(k) = \frac{2\pi}{N_{\mathbb{t}}}\Delta\mathbb{t}(k) \tag{2.84}$$

where $N_{\mathbb{t}}$ represents the number of holes in the wheel and T_s is the sampling time.
All sensor data are utilized to determine the motor actions based on the WMR's objectives. The WMR is equipped with four 5V DC motors and one Tower Pro Micro Servo SG90. These components are responsible for its movement and for rotating the ultrasonic sensor, respectively. To operate the motors, a Pulse-Width Modulation (PWM) signal is required. Therefore, the controller is tuned to provide the correct duty cycle, ensuring that the motors receive the appropriate voltage within the 0-5 Volts range.
On the software side, the controller is designed and developed using Matlab/Simulink to enable the WMR to move and respond to its surroundings. To simulate the WMR and its environment, the Mobile Robotics Training Library is employed. This library offers various blocks to simulate motors, encoders, line sensors, and the tracks to follow.
Subsequently, the code is converted to C++ using Simulink Coder and uploaded to the Nucleo board for serial communication, as illustrated in Fig. 2.16.

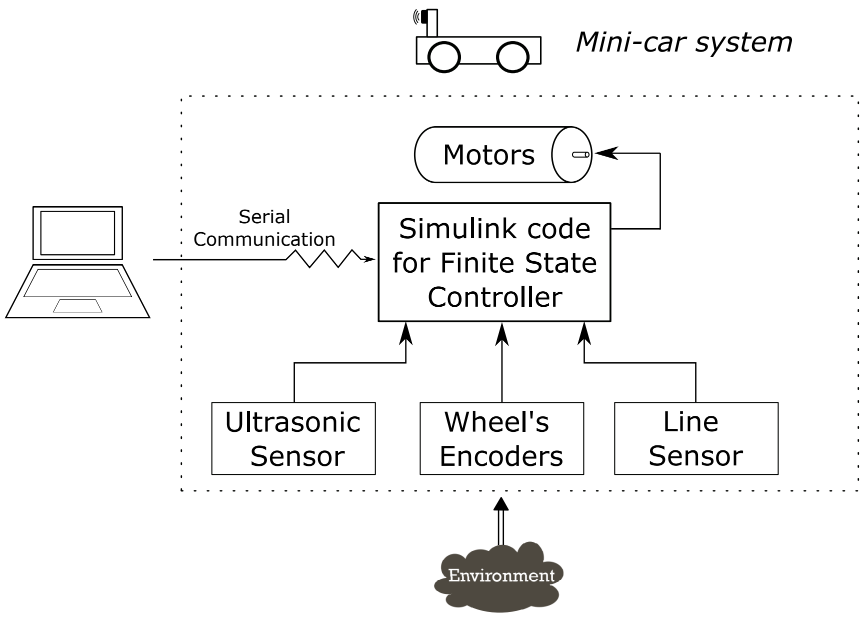


Figure 2.16: Scheme of the system

The analyzed driving scenarios involve a three-lane road with fixed obstacles. The vehicle initiates its journey in the rightmost lane and changes lanes only when compelled to do so due to an obstacle ahead. It is assumed that at least one lane remains obstacle-free for a sufficiently long stretch, allowing the vehicle to settle in that lane before changing lanes to avoid subsequent obstacles.
Both scenarios feature a three-lane map with obstacles positioned on different lanes, influencing the WMR's behavior:

Scenario 1: Zig-zagging
Three obstacles are positioned at distances of 0.7, 1.5, and 2.4 meters from the starting point, which is at the beginning of the rightmost lane. The first and last obstacles are located in the rightmost lane, while the second obstacle is in the center lane. This configuration is expected to result in a zig-zag movement, starting from the right lane and concluding in the center lane.

Scenario 2: Double left lane change
This scenario involves four obstacles: one in the right lane at 0.7 meters, two at 1.5 meters (leaving the left lane free), and the last obstacle at 2.2 meters in the left lane. The expected result is a double left lane change in response to obstacles in the other lanes, followed by repositioning in the center lane.

These two scenarios ensure the replication of every possible maneuver in a three-lane road environment.
Tests have been conducted in both simulated and real-world environments, using a car kit that closely mimics the behavior of an unmanned vehicle. The objective was to compare performance to validate the theoretical model's accuracy and the physical robot's reliability. In both simulation and real-world tests, a flowchart has been designed to outline the vehicle's behavior over time. Within the flowchart, certain states require the robot to perform specific actions, such as following the road, while continuously monitoring the environment for obstacles ahead. When an obstacle is detected, the vehicle checks the availability of other lanes and moves to one of them if necessary. To simplify the flowchart while ensuring a realistic scenario, the following assumptions have been made:

- when in an external lane and encountering an obstacle, it is assumed that the central lane is free.
- when in a central lane and encountering an obstacle, it is assumed that at least one of the external lanes is free. If both are free, priority is given to the rightmost lane.

The resulting event-based controller is visible in Fig. 2.17.

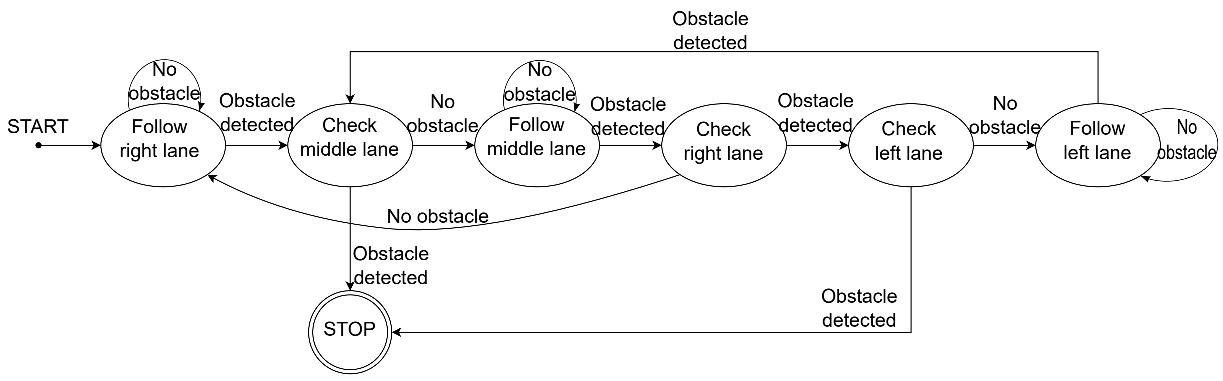


Figure 2.17: Scheme of robot's evolution overtime

A PID controller regulates the voltage supplied to the motors based on input from the line-tracking sensor. It processes the data received from the sensor, as outlined in Table 2.5. The controller uses a reference value of 0, indicating that the center optical sensor detects a black surface.

The choice of a PID controller is due to its simplicity and repeatability, aligning with the minimum lane-keeping requirements for validating the approach in the considered scenario. Given that the vehicle's behavior on a straight lane requires relatively minor control adjustments, a basic controller suffices, providing quick and reliable performance without extensive tuning of its constants. Trial-and-error tuning proved adequate to obtain reliable values for keeping the vehicle on track. Sub-optimal gains remain effective for this type of control, partly because standard tuning methods, such as the Ziegler-Nichols method detailed in [109], necessitate system response measurements for coherent gain adjustments, which were not feasible in the chosen configuration.

The lane change is based on the following discrete-time equation of the evolution of the WMR's direction:

$$\theta(k+1) = \theta(k) + \frac{\Delta D_R(k) - \Delta D_L(k)}{L} \tag{2.85}$$

where θ is the orientation of the WMR, $D_R(k)$ and $D_L(k)$ are the traveled distance by the right and left wheel at time k, and L is the distance between the wheels.

The motors mounted on the WMR have specific operational constraints that necessitate a minimum power input for functioning. To address this limitation, a proportional controller with a bounded output has been implemented.

In comparing a virtual and real case study, minor variations in gain tuning are predictable, due to the distinctive physical characteristics of the WMR. These modifications primarily impact the derivative component of the controller, as summarized in Table 2.6. It is worth noting that alternative values are viable without compromising the overall objective. However, empirical tests have demonstrated that higher derivative term values in the simulated environment can lead to the robot coming to a halt and oscillating as it approaches a lane.

Conversely, the execution of the lane-changing maneuver relies on a high-gain proportional controller coupled with a constraint that halts the rotation of the WMR at a predefined angle, approximately 45°. The utilization of high gain is imperative to accommodate the physical constraints of the WMR, as it requires a minimum power input to overcome friction and initiate rotation.
The inputs of the flowchart are measurements retrieved from sensors, while its outputs are the pulse width modulation to supply to the motors that command the velocity and the rotation of the robot, together with the heading of the servomotor linked to the ultrasonic sensor.

	Kp	Ki	Kd
Real	65	1	2
Virtual	65	1	0.2

Table 2.6: Values of the PID in the real and virtual simulation

2.6.0.1 Virtual tests

Tests in the virtual environment were conducted using Simulink, which provides built-in toolboxes for simulating the movement of autonomous robots. Although the model used in the virtual environment represents a simplified unicycle, it serves as a valuable approximation for predicting the behavior of the autonomous vehicle before implementing the control algorithm on the physical robot.
Specifically, the Matlab Mobile Robot Toolbox has been employed, including the Mobile Robotics Training Library, which provides insights into the model's architecture and design of Eq. 2.80. Sensors are simulated by exploiting Simulink blocks that reproduce the main properties of each WMR component. One notable difference compared to the physical device is that the virtual ultrasonic sensor must align with the vehicle's heading. Consequently, when the sensor rotates in the simulation, it results in the entire robot rotating. However, this adjustment does not impact the overall performance. Fig. 2.18 offers an overview of the WMR's behavior in a simulated environment, confirming its effective execution of the desired maneuvers within lanes. Additionally, Fig. 2.19 illustrates the heading of the vehicle during simulation. It is important to note that the comparison between the heading in the virtual scenario and the real one can only be done approximately, due to the absence of a gyroscope on the WMR to obtain precise orientation measurements.

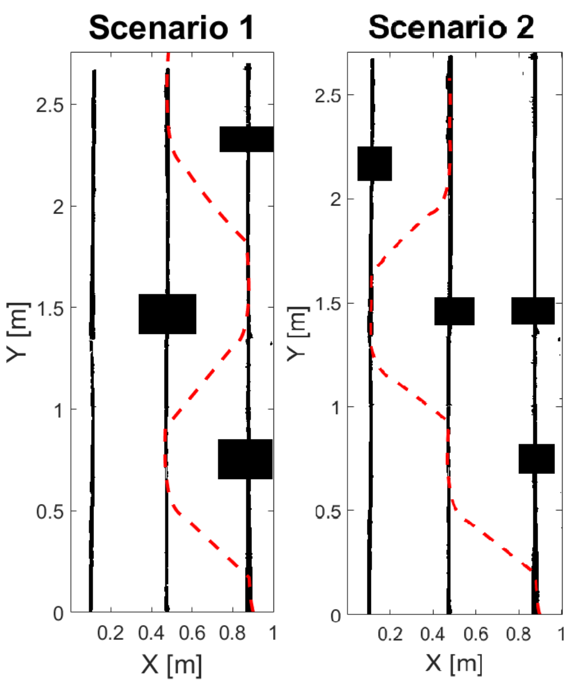
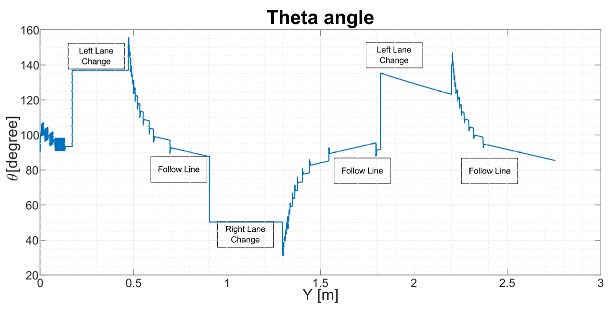
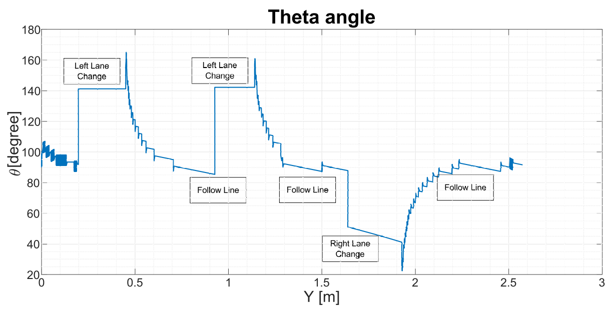


Figure 2.18: Simulation of both scenarios: the zig-zag scenario on the left and the double lane change scenario on the right. The red dotted line is the path of the WMR.



(a) θ angle in the first scenario



(b) θ angle in the second scenario

Figure 2.19: WMR's orientation during simulation in both scenarios

2.6.0.2 Real-world tests

To conduct tests with the physical vehicle, a longitudinal track was designed, consisting of three lanes drawn on a white sheet of paper using a black spray can. This choice, with respect to the common practice of using black tape to draw the road, reduces friction between the wheels and the floor, thereby enhancing the reliability of the WMR's performance.

To represent obstacles, boxes were placed on the road, with a longitudinal separation of at least half a meter. This separation facilitates the vehicle's adjustment after a lane change. The vehicle operated at 40% of its maximum speed, approximately $1m/s$, with a critical obstacle detection distance set at 0.5 meters. This distance was found to be suitable for the ultrasonic sensor to operate with high precision and for the vehicle to have sufficient space for smooth obstacle avoidance.

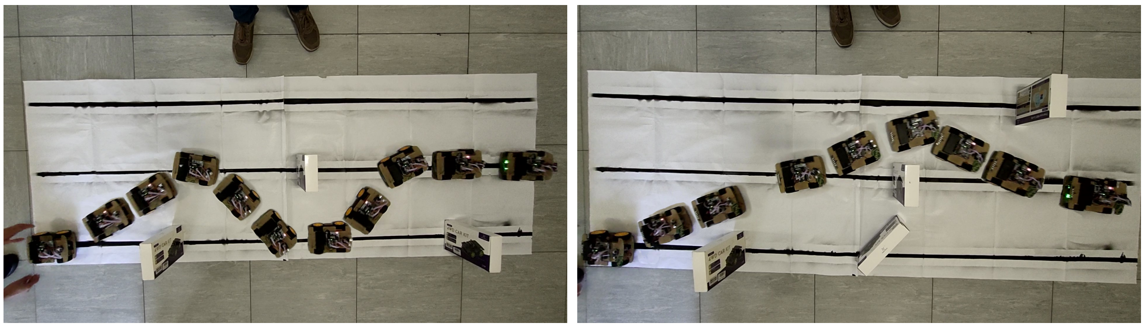

(a) Sequence of zig-zagging (b) Sequence of double lane change

Figure 2.20: WMR real-time behavior in both scenarios

As expected, the real-world implementation introduces dynamics that were not considered in the simulation, primarily due to factors like friction. A photo sequence illustrating the WMR's behavior is shown in Fig. 2.20a. In this scenario, the WMR doesn't complete a full 45° left turn due to higher friction, resulting in a longer path to reach the center lane. Consequently, the WMR detects an obstacle while approaching the target lane, leading to a right turn to change lanes immediately after reaching the center lane. On the contrary, during the second left turn lower friction allows the WMR to turn more effectively, reducing travel time between the left and center lanes, similar to the simulation.
Similarly, Fig. 2.20b illustrates the second scenario, wherein higher initial friction causes the real-world application to exhibit slightly different behavior from the simulation. Initially, the WMR detects the obstacle and initiates a left turn, but friction and wheel slipping cause it to turn less than 45°. This leads to increased travel distance while changing lanes, resulting in the WMR detecting the obstacle in the center lane while still in the process of lane change. Consequently, the WMR starts moving toward the third lane while checking if there's space to return to the right lane. Even before completing the lane change, the last obstacle in the left lane is detected, prompting a midway right turn and stabilization in the center lane.
In summary, the real-world implementation of the WMR highlights the significance of considering friction and wheel slipping effects when designing control algorithms for autonomous vehicles, as these factors can lead to deviations from simulated behavior. The WMR exhibits adaptive behavior in response to these real-world dynamics, showcasing its

ability to navigate through obstacles and change lanes effectively, even if a deeper understanding of its dynamics is needed to improve the reliability of its driving.

Chapter 3

Routing in Smart Manufacturing Systems

In the preceding chapter, the exploration of autonomous vehicles and platooning unveiled the technologies revolutionizing transportation and logistics. The principles of efficient control and coordination extend beyond transportation and apply to a broader spectrum of domains.

This chapter shifts the focus to the core of modern production: smart manufacturing systems. Here, the significance of routing and scheduling, disciplines that manage machine operation and task allocation in the production chain, becomes evident. This chapter analyzes some techniques in manufacturing scheduling, emphasizing its role in optimizing resource allocation and production processes, before bridging these concepts in the following chapter with a case study that demonstrates the integration of platooning-based control techniques and AGVs in smart manufacturing systems.

The principles explored in autonomous vehicle platooning provide a foundation for the coordination and control required in manufacturing. AGVs represent the link to connect the two domains, as they can be considered dynamically similar to autonomous vehicles for what concerns their displacement within the environment. In that direction, platooning-based control techniques applied to AGVs in smart manufacturing systems are treated in the next chapter. This integration exemplifies how the principles of autonomous vehicle platooning can enhance manufacturing processes, promoting efficiency and adaptability within the context of smart factories.

However, before handling the trajectories, it is necessary to consider the allocation of products (or jobs) through machines. It involves determining which machine should perform each task, considering factors such as machine capabilities, processing times, and task priorities.

Various types of scheduling problems are employed based on their characteristics, desired requirements, and the type of manufacturing system. Each of these scheduling types has a corresponding scenario in transportation:

- **Open Shop Scheduling**: In an open shop, each job consists of a sequence of operations that can be processed on any available machine without any specific order. It offers the highest level of flexibility in terms of machine assignment, making it suitable for job sequencing where machine order doesn't matter. Solving open shop scheduling problems can be challenging due to the high level of flexibility, making it computationally complex. However, it can be commonly found in industries where various jobs have different process requirements and can be completed on multiple machines in any order. In transportation, a similar concept exists in the road environment, where vehicles may merge into a platoon when they share a common route passing through the same waypoints.

- **Flow Shop Scheduling**: In a flow shop, each job follows a fixed sequence of operations, and all jobs must follow the same order on the machines. Flow shops are more rigid in terms of job sequencing, as all jobs must adhere to the same predefined sequence. On the other hand, they are efficient for repetitive and standardized manufacturing processes with limited job variations and are commonly used in mass production, where products follow a consistent assembly line. The concept of flow shop scheduling aligns with public transportation systems, such as buses and trains. Indeed, each vehicle follows a fixed and predefined route, and all vehicles must adhere to the same order of stops, analogously with jobs following a fixed sequence through machines in manufacturing. Moreover, buses sharing some of their stops along the route may merge into platoons to enhance performance.

- **Job Shop Scheduling**: Job shops are characterized by high variability, where each job has a unique sequence of operations and can be processed on different machines with no specific order, offering high versatility to handle customized or low-volume production, as jobs are tailored to specific customer requirements. Solving job shop scheduling problems can be highly complex due to the wide range of possible job sequences and machine assignments and their usage involves sectors in which products are made-to-order and diverse, such as custom manufacturing. A known variant is the permutation job shop scheduling, in which each job has a set of valid sequences of operations, and any one of these sequences can be selected for the job, with the final goal of determining the assignment of jobs to machines and the selection of a specific sequence for each job while optimizing specific criteria. An analogous concept in the road environment can be found in logistics and delivery services. Each company has its own delivery routes and trucks can form platoons to go from the warehouse to retailers in string formation.

- **Hybrid Shop Scheduling**: Hybrid shops combine elements of open shop, flow shop, and job shop scheduling based on the specific requirements of the manufacturing process, providing a balance between flexibility and structure, and allowing for customized sequences and machine assignments when needed. In the road environment hybrid shop scheduling resembles ridesharing, with scheduled routes for regular passengers and the flexibility to provide on-demand services with varying routes and destinations. However, there are differences between hybrid shop scheduling and ridesharing, particularly since in the latter platoons do not currently exist, despite their potential applicability on frequently used routes.

3.1 Flexible job-shop scheduling problem

In a flexible job-shop scheduling problem (FJSP), the objective is to allocate J jobs on M machines to execute O_j operations each ($j = 1, ..., J$). The number of operations for each job may differ and cycles (i.e. multiple passage of jobs on machines) may be present in their path, increasing the general complexity of finding a feasible and efficient solution. With the advancement of technology, in recent years the evolution in manufacturing has given birth to the flexibility of machines. In other words, certain machines are able to accommodate multiple operation types, thereby enabling distinct routing possibilities within the shop for each job. In the following, all the possible paths of a job within the shop are enumerated, providing a range of alternatives, namely A_j, representing the choices for job j, collectively forming the set $A = A_1, A_2, ..., A_J$. This augmentation amplifies the system's adaptability and flexibility.
The interplay between jobs' routings can give rise to shared resources, representing the machines through which multiple jobs have to pass during the production process. This leads to the formulation of sets of disjunctive connections, denoted as D, which ensures the orderly processing of one job at a time on shared machines. This dynamic interplay is illustrated in Fig. 3.1, where a generalized graph G visually captures the product flow through the shop. Alternatives are organized into rows by job, signifying potential sequencing on machines. It's worth noting that precisely one alternative must be selected for each job. The diagram further highlights disjunctive connections, such as δ_1 and δ_2, demonstrating the interconnections between alternatives of different jobs on shared machines (e.g., $M^{i+1}_{a_{1,1}} = M^{i+1}_{a_{j,1}}$ and $M^{i+1}_{a_{j,A_J}} = M^{N}_{a_{1,A_1}}$).

3.1.1 Mathematical formulation of the FJSP

In order to systematically address the optimization problem through mixed integer linear programming (MILP), it is essential to establish a clear framework by introducing the relevant sets, variables, and constants that define the key parameters governing the system.

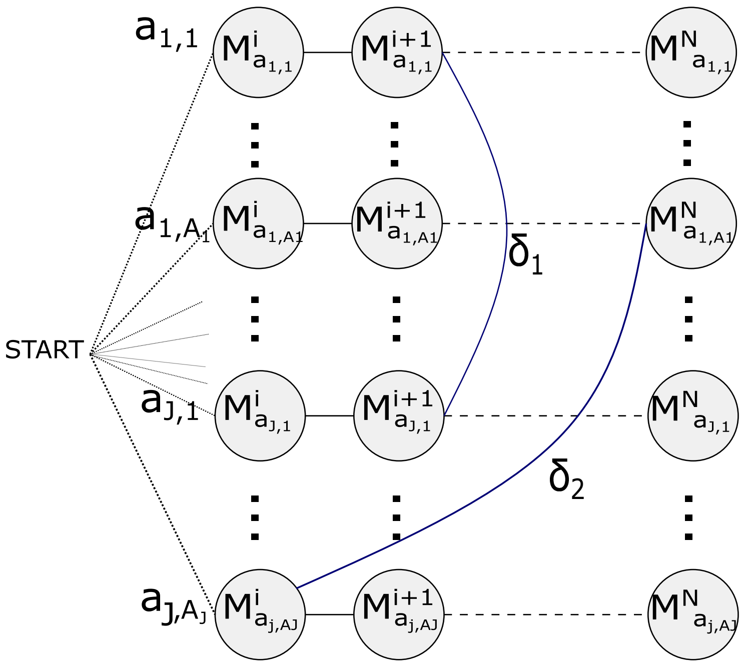


Figure 3.1: Flow-shop graph representation of the FJSP

Therefore, the sets are:

- J: number of jobs;
- M: number of machines;
- D: number of disjunctive connections on shared machines;
- A: number of alternatives;
- O_j: number of operations for each job j.

Meanwhile, the decision variables of the problem are:

- $s_{j,m} \in \mathbb{R}$, $[J \times M]$: the start time of job j on machine m;
- $c_{j,m} \in \mathbb{R}$, $[J \times M]$: the completion time of job j on machine m;
- $\delta_d \in \{0,1\}$, $[D \times 1]$: variable for disjunctive connections between shared resources;
- $\gamma_{j,a_j} \in \{0,1\}$, $[A \times 1]$: variable for modeling possible job's choices;
- $C \in \mathbb{R}$, $[1 \times 1]$: completion time of the last job that completes the production process, also known as makespan.

The constants of the problem are:

- P, $[J \times M]$: matrix of ideal processing times, where $p_{j,m}$ corresponds to the processing time of job j on machine m. Processing times can be affected by disturbances that cause delays in production;
- $\mathbb{M}$: big-M, used to activate and deactivate pairs of constraints;
- R_0, $[J \times 1]$: vector of jobs' release time.

The associated mixed integer programming problem minimizes the completion time of jobs:

$$\min C \tag{3.1}$$

subject to constraints expressed in Eqs. 3.2.a-3.2.g:

$$s_{j,m(j,o)} \geq c_{j,m(j,o-1)} - (1-\gamma_{j,a_j})\mathbb{M} \qquad \forall j = 1,..,J \quad \forall o = 2,...,O_j \tag{3.2.a}$$

$$c_{j,m} = s_{j,m} + p_{j,m}\sum_{a_j=1}^{A_j}\gamma_{j,a_j} \qquad \forall j = 1,..,J \quad m = 1,...,M \text{ and } a_j = 1,..A_j \tag{3.2.b}$$

$$s_{j_1,m} \geq c_{j_2,m} - \delta_d\mathbb{M} \qquad \forall j_1, j_2 \in D \tag{3.2.c}$$

$$s_{j_2,m} \geq c_{j_1,m} - (1-\delta_d)\mathbb{M} \qquad \forall j_1, j_2 \in D \tag{3.2.d}$$

$$s_{j,m(j,1)} \geq R_0 \qquad \forall j = 1,..,J \tag{3.2.e}$$

$$C \geq c_{j,m(j,O_j)} \qquad \forall j = 1,..,J \tag{3.2.f}$$

$$\sum_{a=1}^{A_j}\gamma_{j,a,j} = 1 \qquad \forall j = 1,..,J \tag{3.2.g}$$

Specifically, Eq. 3.2.a establishes the relationship between the start and completion time of consecutive machines in the sequence of job j, conditioned to the selected alternative. The notation $m(j,o)$ serves to denote the machine that processes job j at its o^{th} operation in the sequence. Eq. 3.2.b ensures the adherence to the correct processing time within each machine. The disjunctive constraints are represented in Eqs. 3.2.c-3.2.d, where the δ_d variables guarantee the activation of only one of the two constraints for each shared machine. Furthermore, Eq. 3.2.e ensures compliance with the job release times in the shop, while Eq. 3.2.f determines the completion time of the last processed job. Lastly, Eq. 3.2.g addresses the γ variables, ensuring that for each job, a single alternative is chosen, thereby avoiding multiple paths for the same job.
In summary, the optimization problem aims to determine the optimal routing to minimize job waiting times and completion time, while adhering to resource availability, machine occupancy, and job processing flow constraints.

3.1.1.1 Handling of unknown processing times

In the preceding section, deterministic processing times were assumed in the system. However, in general, they are subject to disturbances that may delay the completion time. Thus, it is needed to account for real-world variability and provide a valuable solution.
To ensure a realistic behavior of the system, two key assumptions have been taken into account: each machine can, at most, double its processing time, and the intensity of disturbances that affect the system is bound by an upper limit denoted as Ω. To achieve this, an iterative algorithm has been designed to strike the optimal balance between a solution unaffected by noise and a resilient solution capable of handling worst-case scenarios, in which disturbances on machines maximize the delay in production. Thus, the possible delay of job j on machine m is represented with a new decision variable $w_{j,m}$.
The optimization cost function then translates into a minimax problem: the former minimization problem, which focused on reducing the makespan, has evolved into a minimization of the makespan while accounting for the maximum delays, representing the most adverse machine delay scenarios:

$$\min \max_{\underline{\omega}} C \tag{3.3}$$

Constraints in Eqs. 3.2.a-3.2.g remain valid, except for Eq. 3.2.b which needs to be extended to accommodate delays and be independent of γ:

$$c_{j,m} = s_{j,m} + p_{j,m} + (p_{j,m}\omega_{j,m}) \quad \forall j = 1,..,J; m = 1,...,M \text{ and } a_j = 1,..A_j \tag{3.2.b'}$$

Then, additional constraints are needed to realistically model delays:

$$\sum_{j=1}^{J}\sum_{i=1}^{A_j}\gamma_{i,j}\sum_{m=1}^{M}\omega_{j,m} = \Omega \tag{3.4}$$

$$0 \leq \omega_{j,m} \leq 1 \quad \forall\, j = 1,...J \quad m = 1,...,M \tag{3.5}$$

In Eq. 3.4, the optimization problem ensures that only the $\omega_{j,m}$ values associated with the alternatives selected in the previous minimization problem are considered in the computation of the delay configuration. Furthermore, Eq. 3.5 enforces an upper limit on the magnitude of each disruption, ensuring that, in the worst-case scenario, each machine is allowed to at most double its processing time, and not beyond.

3.1.2 Minimum regret with stochastic processing times

Clearly, it is interesting to evaluate the solution in case of delays and for different intensities of disturbance. To this aim, random instances of noises can be generated with different magnitudes (i.e. Ω), leading to different scenarios σ. One feasible solution is empirically found and then X mutations are performed on the primordial solution by imposing one alternative. Then, solutions are evaluated with the minimax regret criterion, in which the regret is chosen with respect to the makespan:

$$R(\sigma, x) = C(\sigma, x) - C^*(\sigma) \tag{3.6}$$

$C^*(\sigma)$ indicates the optimal solution with respect to the objective function of Eq. 3.1 in the scenario σ, while $C(\sigma, x)$ is the completion time of solution x in the same scenario. Then, for each solution, the worst-case scenario (i.e. the scenario that maximizes the regret) is found:

$$R_w(x) = \max_{\sigma} R(\sigma, x) \tag{3.7}$$

Finally, the solution that minimizes regret in the worst-case scenario is chosen as the best solution.

$$R^* = \min(R_w(x)) \tag{3.8}$$

The general idea behind the proposed approach relies on the NP-hardness of the optimal solution, which usually can be hardly found or even estimated. For this reason, it seems valuable to provide feasible solutions by imposing specific paths for some jobs. It may happen within flexible manufacturing systems that, for some external reasons, the path on one job is imposed by physical limitations. This drastically reduces the number of feasible solutions to evaluate and thus the computational burden. In order to enhance the robustness of the system, it is reasonable to evaluate candidates in different scenarios to choose the one that gives the minimum regret in the worst case, especially if the physical constraints on the path of jobs are not restrictive and there is freedom in the choice of the solution.
However, in more complex problems, the optimal solution may be too heavy to compute; therefore it is recommended to evaluate the regret with respect to C_{lb}, the lower bound of the completion time, which can be calculated using the heuristic of [110].

3.1.3 Ensuring robustness with unknown processing times

In order to find a solution capable of keeping competitive performance both in the case of deterministic and unknown processing times, with the only assumption of knowing its overall magnitude Ω, an iterative algorithm can be designed:

Step 1: Solve the problem with deterministic processing times. This solution represents the minimum achievable completion time for the given scenario in the ideal case and can be found by solving the problem in Eq. 3.1-3.2.

Step 2: Solve the problem with stochastic processing times. Set the recently determined γ variables as parameters for the max sub-problem in Eqs. 3.2-3.5, while keeping the δ variables to be determined through the optimization problem. This involves imposing the path for each job while using the sequencing on machines and disturbance on processing time, namely δ and ω, as decision variables. A solution for ω represents the worst-case scenario in the given routing, while still allowing flexibility in sequencing on machines

Step 3: Compare the two solutions and update the best trade-off solution. : Revisit the problem with deterministic processing time, setting $p_{j,m} = p_{j,m} + p_{j,m}\omega_{j,m} \quad \forall j = 1, ..., J; m = 1, ..., M$ and solve the problem anew. In few words, this translates into solving the deterministic problem in the case of the worst possible delays. Subsequently, compare the new job path solution (i.e., only the γ variables) with the one obtained in Step 1:

- If the two solutions coincide, it indicates that the job path represents the best trade-off between optimality and robustness when confronted with potential delays of the given magnitude
- Should the new solution differ from its predecessor, it suggests that the delay-free solution is no longer optimal provided conditions of Eqs. 3.4- 3.5 hold. In such an instance, to derive an appropriate solution for both scenarios, the γ variables of the novel solution are once again set as parameters, and Step 2 is reiterated.

This approach, explained in Algorithm 1, enables the identification of a job path that performs well across a range of Ω values, thus accommodating potential delays, even if guaranteed convergence is not assured. Consequently, the algorithm is intentionally halted after a predetermined number of iterations, with the selection of the solution minimizing the completion time in the deterministic scenario.

Algorithm 1 Algorithm to find a trade-off between the optimality and robustness

```
    Input: P, R        // Processing times, Release times
    Output: Result
 1: solution1 = solve(Eqs. 3.1-3.2, with 3.2.b, P, R)
 2: $\gamma$ = Solution1.$\gamma$        // Set gamma as parameters
 3: solution2 = solve(Eqs. 3.2-3.5, with 3.2.b', P, R, $\gamma$)
 4: if "solution1" == "solution2" then
 5:     Result = solution1
 6:     break
 7: else
 8:     Solutions.append(Solution1) // Add Solution1 to the pool of solutions
 9:     "solution1" = "solution2."
10:     if Max iterations reached then
11:         Result = min(Solutions) // Get the solution with minimum completion time
12:         break
13:     end if
14:     Go to Line 3.
15: end if
```

3.2 Model predictive control-based approach for online scheduling

The problem discussed in Section 3.1.1 pertains to the offline scheduling of jobs through machines. In ideal situations, possessing comprehensive insight into the entire production process becomes crucial to ensure the accuracy of the solution. However, there might be instances where this information is unavailable, or the only accessible data is the planned timing of job releases over time. As a result, the aforementioned problem needs an expansion to tackle the scheduling challenge in real-time fashion, with each new product arrival. In order to accomplish this, it is necessary to keep track of the progress of jobs that have undergone operations within the shop. Thus, at each event denoted as t, marking the arrival of a new product, the γ representing paths that are no longer feasible, due to the job having already completed a portion of the route between machines, rendering its passage through other machines impossible, are removed from the job's potential alternatives. Analogously, the machines currently assigned to jobs are retained until their completion time, to avoid their assignment to other jobs when solving the optimization problem.

$$s_{j,m}^{t} = s_{j,m}^{t-1} \tag{3.9.a}$$
$$c_{j,m}^{t} = c_{j,m}^{t-1} \tag{3.9.b}$$

where t represents the t^{th} event (i.e. arrival of a job) on the shop floor.
These constraints enable the execution of the scheduling algorithm described in the previous section for every arrival of a product. Additionally, a prediction horizon can be incorporated to facilitate the scheduling of jobs currently present on the shop floor and those planned to arrive shortly (i.e. within the prediction window).
The foundations lies in the Model Predictive Control (MPC), a powerful and widely used control strategy in manufacturing processes to optimize the operation of complex systems while adhering to various constraints. Nevertheless, it is hugely used in autonomous driving due to its ability to plan and optimize vehicle control actions over a short prediction horizon while considering dynamic constraints and obstacles, enabling safe and efficient real-time decision-making. In this field, a the mathematical formulation considers an objective function as follows:

$$\min_{U} \sum_{k=0}^{N_p-1} \|y(t+k|t) - r(t+k)\|_Q^2 + \sum_{k=0}^{N_c-1} \|u(t+k|t)\|_R^2 \tag{3.10}$$

where:

- U represents the control input sequence over the prediction horizon;
- $y(t+k|t)$ is the predicted output of the system at time $t+k$ given the current state at time t;
- $r(t+k)$ is the reference trajectory for the system at time $t+k$;
- N_p and N_c are respectively the prediction and control horizon, which specify how far into the future the system's behaviour is predicted and how many control input are optimized at each time step;
- Q and R are weight matrices that determine respectively the importance of tracking the output trajectory and controlling the inputs.

System's dynamics are modeled through the classical equation:

$$x(t+k+1) = f(x(t+k), u(t+k)) \tag{3.11}$$

where $x(t+k+1)$ represents the state of the system at tike $t+k+1$ and $f(\cdot)$ is the dynamic model of the system, describing how the system evolves over time based on its current state and control inputs. It has to be noted that the formulation allows for non-linear systems. Lastly, between the advantages of MPC, there is the possibility of taking into account constraints such as:

- Input constraints: $u_{min} \leq u(t+k|t) \leq u_{max}$

- Output constraints: $y_{min} \leq y(t+k|t) \leq y_{max}$
- State constraints: $x_{min} \leq x(t+k|t) \leq x_{max}$
- Rete of change constraints: $\Delta u_{min} \leq \Delta u(t+k|t) \leq \Delta u_{max}$
- and many more

In MPC scheduling for manufacturing, the optimization problem is solved at each event (i.e. arrival of one or more job), considering only the jobs currently present in the shop floor, and the prediction horizon helps in considering jobs that are supposed to arrive soon in the future. Consequently, the set J, which represents the set of jobs considered in the optimization problem, is initially a subset of the complete job pool. As the final job arrives on the shop floor, this set J encompasses the entire spectrum of jobs.
Thus, for each new event:

$$\begin{aligned} &\forall \mathrm{p} \in \mathcal{P}, \forall \mathrm{s} \in \mathcal{S}\, (\mathrm{s}_\gamma \cap \mathrm{p}_\gamma = \emptyset, \mathcal{P}_\gamma \subseteq \Gamma) \\ &\quad with \quad \#\, (\mathcal{S} \cup \mathcal{P}) = J \end{aligned} \tag{3.12}$$

where

- $\mathcal{P}$ contains all the jobs within the prediction horizon;
- p is a job belonging to set $\mathcal{P}$;
- $\mathcal{S}$ contains all the jobs that have been already scheduled or are being processed;
- s is a job belonging to set $\mathcal{S}$;
- Γ is the set of all alternatives;
- Pedix γ refers to the chosen alternative for the job of the corresponding set.

This translates into a dynamic, reactive controller capable of scheduling jobs based on the system's current state while forecasting the imminent arrival of new products. Fig. 3.2 graphically represents the implemented algorithm.

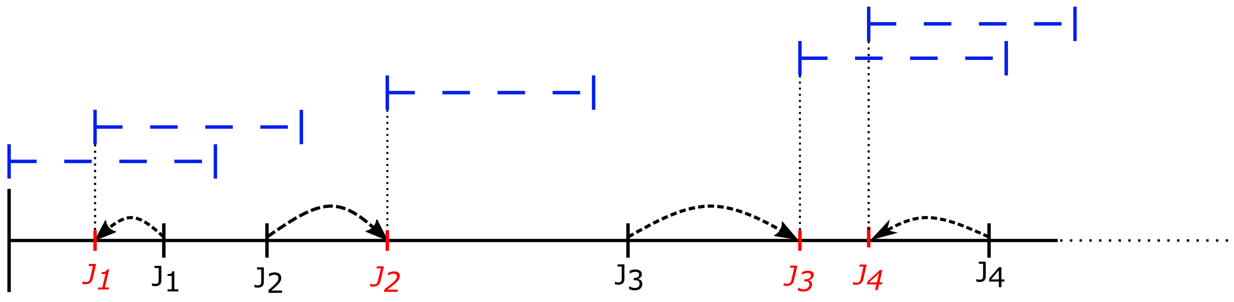


Figure 3.2: Graphical representation of the MPC-based scheduling. Black dashes on the timeline represent the planned arrival time of jobs, and red dashes their real arrival time. Blue lines indicate the prediction horizon of the algorithm, triggered at each new arrival of a job and needed to figure out which jobs are considered in the scheduling problem.

3.3 Offline scheduling: AIP-PRIMECA application

PRIMECA stands for "Pôles de Ressources Informatiques pour la Modélisation et l'Enseignement en Conception Assistée" in French, which translates to "Computer Resources Centers for Modeling and Teaching in Computer-Aided Design". PRIMECA networks are communities of expertise and resources designed to support teaching and research in these fields. The AIP-PRIMECA ("Atelier Inter-Etablissement de Productique dans PRIMECA", which can be translated as "Inter-Institutional Workshop for Production Engineering in PRIMECA") production cell in Valenciennes, France, serves as a benchmark [111] for the methodologies discussed in previous sections. This production cell comprises seven distinct machines:

- **M1**: Loading/Unloading Unit - Responsible for loading plates at the beginning of the production process and unloading finished products at the end. All products initiate and conclude their production cycle on this machine.
- **M2,M3,M4**: Assembly Workstations - These stations perform the necessary production operations to complete a product. For simplicity, a subset of the possible operations has been considered and they can perform common operations in pairs, i.e.
  - M2 and M3 can handle OP1
  - M2 and M4 can handle OP2
  - M3 and M4 can handle OP3
- **M5**: Automatic Inspection Unit - Analyzes finished products to detect any potential faults.

- **M6**: Recovery Unit - This is the sole manual workstation within the system. It is assumed that the recovery unit can repair any product defects.
- **M7**: Optional Workstation - Used primarily in dynamic scenarios with complex product routing through the machines. It effectively represents a copy of M2.

Without loss of generality, the production cell can be simplified by assuming no faults, deleting the optional workstation, and reducing the types of assembly operations. Thus, the machines are reduced to five and for sake of completeness, their deterministic processing time is expressed in Table 3.1, where the absence of a number indicates the inability of the machine to perform that operation.

Table 3.1: Processing time

	OP_L	OP_1	OP_2	OP_3	OP_I	OP_U
M_1	10	-	-	-	-	10
M_2	-	20	20	-	-	-
M_3	-	-	30	30	-	-
M_4	-	20	-	20	-	-
M_5	-	-	-	-	10	-

A simple production chain of three different products is designed, with their operations shown in Table 3.2. Setup and transportation time between machines are supposed negligible, and all products are available to be dispatched on machines from the beginning.

Table 3.2: Sequence of operations for each job

J_1	OP_L	OP_1	OP_2	OP_I	OP_U
J_2	OP_L	OP_1	OP_3	OP_I	OP_U
J_3	OP_L	OP_2	OP_3	OP_I	OP_U

The overall graph in Table 3.3 lists all the possible paths for each job.

Table 3.3: Flexible job-shop schematization, with jobs path for each γ

Job	Choice	O_1	O_2	O_3	O_4	O_5
J_1	γ_1	1	2	2	5	1
	γ_2	1	2	3	5	1
	γ_3	1	4	2	5	1
	γ_4	1	4	3	5	1
J_2	γ_5	1	2	3	5	1
	γ_6	1	2	4	5	1
	γ_7	1	4	3	5	1
	γ_8	1	4	4	5	1
J_3	γ_9	1	2	2	5	1
	γ_{10}	1	2	4	5	1
	γ_{11}	1	3	2	5	1
	γ_{12}	1	3	4	5	1

3.3.1 Evaluation of minimax regret for AIP-PRIMECA

The technique explained in Section 3.1.2 is employed on the AIP-PRIMECA production cell, for $\Omega = \{1, 4, 7, 10, 13\}$. Since the problem has a bearable complexity, the optimal solution is found for each magnitude of the set, from the smallest one to one of the biggest possible (given that $\Omega \leq 15$).

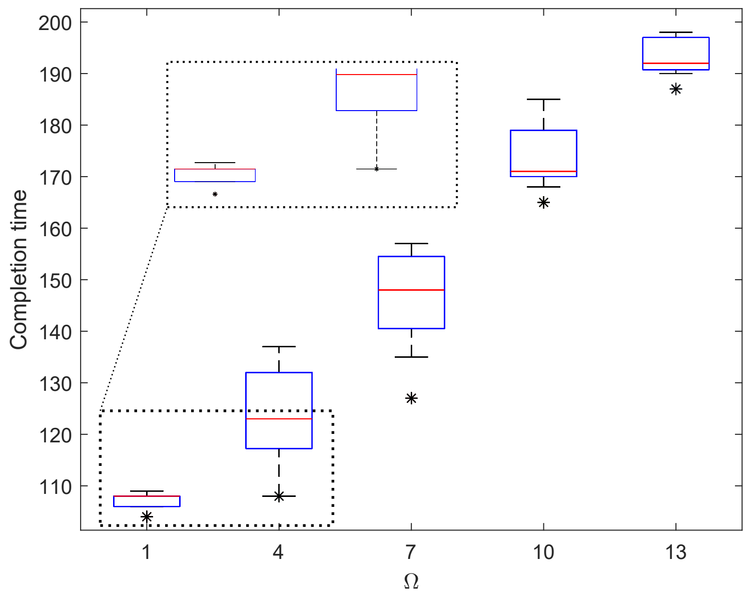


Figure 3.3: Boxplot of the robust analysis. The zoomed area points out that all the solutions lie above the optimal one, guaranteeing the correctness of the optimization algorithm.

Fig. 3.3 shows the data distribution over the different values of Ω, with the black point that indicates the optimal value of C for the given scenario. It is evident that all solutions lie above the optimal one, proving the correctness of the algorithm. Table 3.4 exhibits each solution's selected alternatives (i.e. γ). This imposes the path on machines but still preserves some degree of freedom with the sequencing choice, determined by the δ. Forcing some sequencing constraints represents a further refinement of the algorithm but it increases considerably the complexity of finding a feasible solution and may easily lead to unfeasibility. The proposed approach strikes a good trade-off between the simplicity of finding a pool of feasible solutions and its reliability compared to the optimal one in different scenarios.

Table 3.4: Solutions compared in the regret minimization

Solution	1	2	3	4	5	6	7	8	9
Path (γ_i)	1-8-11	2-8-11	3-8-11	4-8-9	3-5-10	1-6-12	1-7-12	2-8-10	1-8-12

Fig. 3.4 shows the regret values each solution has for different intensities of Ω. Surprisingly, greater magnitudes do not necessarily imply greater regret. This occurs because, when the optimal solution is significantly affected by delays, other feasible solutions may converge to similar completion times due to the system flexibility. It is evident that solution 6

exhibits the minimum regret even in the worst-case scenario, significantly outperforming other values. Therefore, it represents a robust and valid alternative to the optimal solution, ensuring good performance and high reliability for various delay scenarios. Additionally, it is worth noting that solution 9 is the optimal one for $\Omega = 4$, but it performs poorly compared to other feasible solutions in different scenarios.

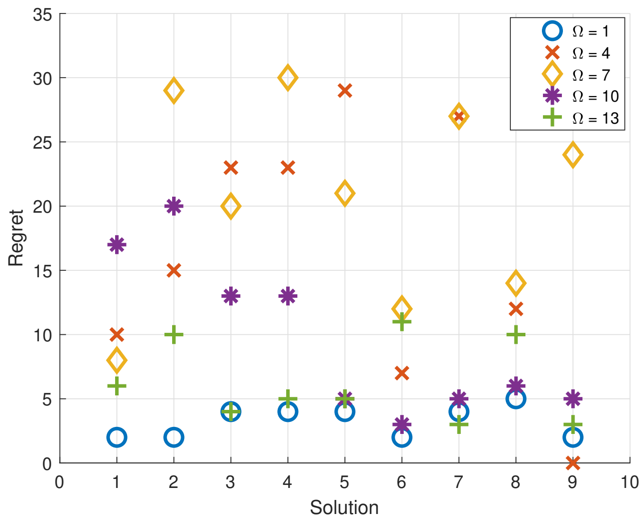
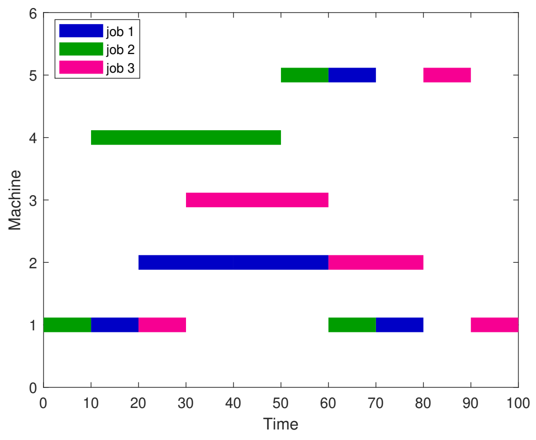


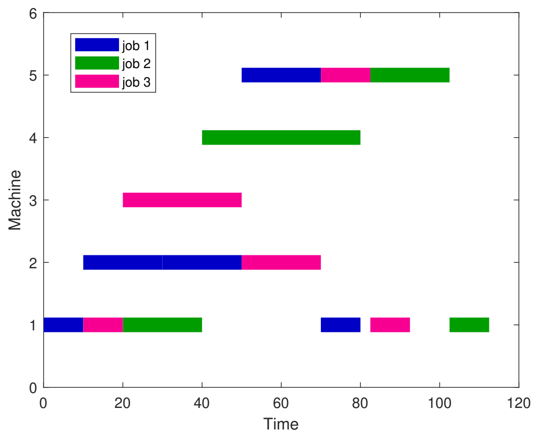
Figure 3.4: Regret values for each feasible solution and each distinct value of Ω, depicted with distinct symbols to have a visually more appealing representation.

3.3.2 Evaluating robustness of solution for AIP-PRIMECA

The iterative algorithm in Section 3.1.3 is analyzed to prove its effectiveness on the same AIP-PRIMECA production cell [112].
The optimal solution, without disturbances, is depicted in Fig. 3.5a and represents the minimum achievable value when minimizing the completion time $C^* = 100$.



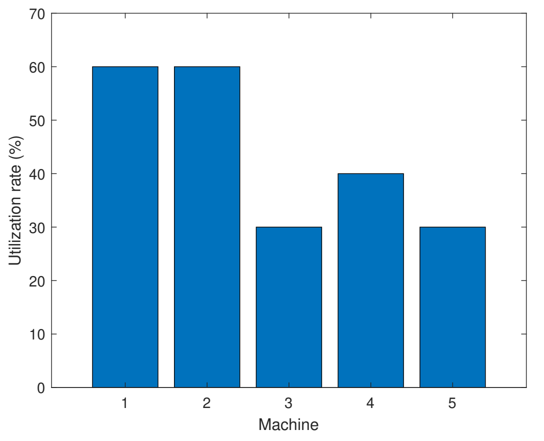
(a) Scheduling solution in absence of disturbances



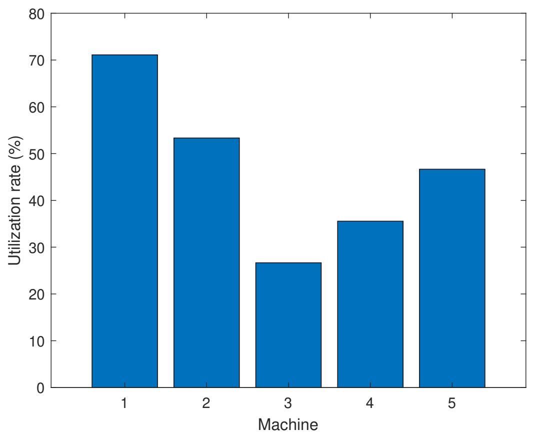
(b) Scheduling solution in the case of $\Omega = 7$

Figure 3.5: Comparison of noise-free and disturbed solutions. Completion time keeps competitive performance even with highly disturbed processing times.

Additionally, $M1$ can be expected as the main bottleneck of the system, as each product needs to start and finish with an operation on that machine. This is confirmed by Fig. 3.6a, which also points out how the slowest machine ($M3$) is, reasonably, the least used, as other machines can perform the same operations in a shorter amount of time.



(a) Machines utilization rate in absence of disturbances



(b) Machines utilization rate with $\Omega = 7$

Figure 3.6: Comparison of noise-free and disturbed machine utilization. The algorithm prevents over-utilization of specific machines by balancing the load.

Increasing the magnitude of disturbances provokes changes in the scheduling/routing of the jobs, as shown in Table 3.5. To have a deeper insight, Fig. 3.5b points out the differences in scheduling when $\Omega = 7$. It can be seen that, in order to keep an acceptable completion time also with the presence of disturbances, the optimization algorithm performs changes in the scheduling of the jobs. Even if there are some hard constraints in jobs routing, the algorithm still has some freedom in choosing the sequencing through machines and this allows not to degrade performances. Moreover, Fig. 3.6b shows the uniform utilization of machines to limit the risk of fault as an emergent behaviour. A slightly higher utilization of $M1$ is noticeable, but it is due to the delays in its processing time and it can not be avoided, as it is the load/unload machine and a necessary step in all jobs production chain. However, the algorithm provides a solution that balances the usage of the assembly machines.

Table 3.5: Comparison of solutions for increasing values of Ω

Ω	Jobs path	Start order	C^*
1	1-2-2-5-1 1-4-4-5-1 1-3-2-5-1	$J_2 - J_1 - J_3$	105
3	1-2-2-5-1 1-4-4-5-1 1-3-2-5-1	$J_2 - J_1 - J_3$	110
5	1-4-2-5-1 1-4-3-5-1 1-2-2-5-1	$J_3 - J_1 - J_2$	110
7	1-2-2-5-1 1-4-4-5-1 1-3-2-5-1	$J_1 - J_3 - J_2$	112.5
9	1-2-2-5-1 1-2-3-5-1 1-3-4-5-1	$J_1 - J_3 - J_2$	130
11	1-2-2-5-1 1-4-4-5-1 1-3-4-5-1	$J_1 - J_2 - J_3$	130

To additionally test the robustness of the solution, 1000 iterations have been launched for each of the chosen Ω with fixed path, fixed sequencing, and random disturbances, generated with the aid of [113] in order to meet constraints of Eqs. 3.4-3.5. Data distribution is visible in Fig. 3.7 as further evidence of the correctness of the solution found, as all other values have a longer completion time.

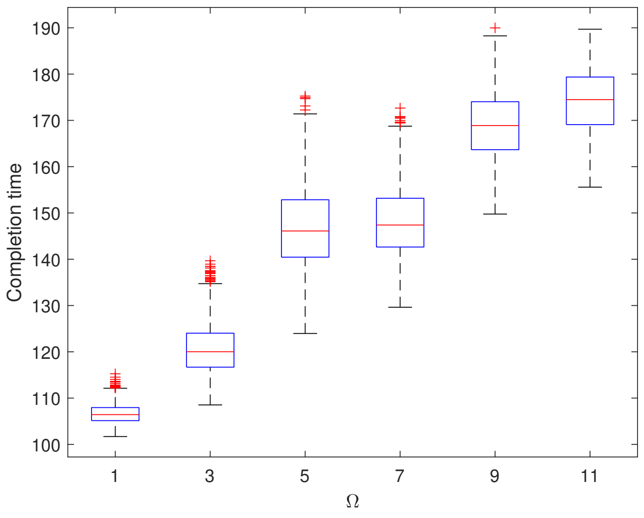


Figure 3.7: Data distribution of 1000 simulations for each Ω

3.4 Online scheduling: evaluation of robust MPC-based approach

The performance of the dynamic MPC-based scheduling of Section 3.2 is compared with the offline solution to present a case study with dynamic adaptation to the production chain. Offline scheduling typically yields faster completion times but demands a comprehensive knowledge of the system and longer computational efforts. In contrast, dynamic scheduling can adapt to job release times that deviate from the initial plan, reconfiguring schedules with each new arrival. Consequently, it only requires information about the jobs currently in the shop and those expected to arrive shortly (i.e. within the prediction horizon).
To compare the two approaches, a theoretical case study is devised, involving $M = 6$ machines and $J = 6$ jobs. The complete range of feasible paths is outlined in Table 3.6, in which each machine is represented as a number for sake of notation. The table highlights that all jobs are required to commence from M_1, which serves as the loading unit of the SMS. Furthermore, it is notable that jobs have to perform from 4 to 5 operations, and certain alternatives may share segments of the initial path, thereby enhancing the MPC-based scheduling's ability to dynamically select the optimal route depending on the real-time release of other jobs.

Table 3.6: Alternative paths for each job

Job	Alternative	O_1	O_2	O_3	O_4	O_5
J_1	γ_1	1	2	3	4	5
	γ_2	1	3	5	6	
J_2	γ_3	1	2	3	4	6
	γ_4	1	2	4	5	
J_3	γ_5	1	2	5	6	
	γ_6	1	2	3	5	
	γ_7	1	3	4	6	
J_4	γ_8	1	2	6	5	3
J_5	γ_9	1	4	6	5	3
	γ_{10}	1	2	4	5	6
J_6	γ_{11}	1	3	4	5	
	γ_{12}	1	2	3	4	6
	γ_{13}	1	3	4	6	

The flexibility of machines is expressed by Table 3.7, while the planned and real release time of jobs is listed in Table 3.8.

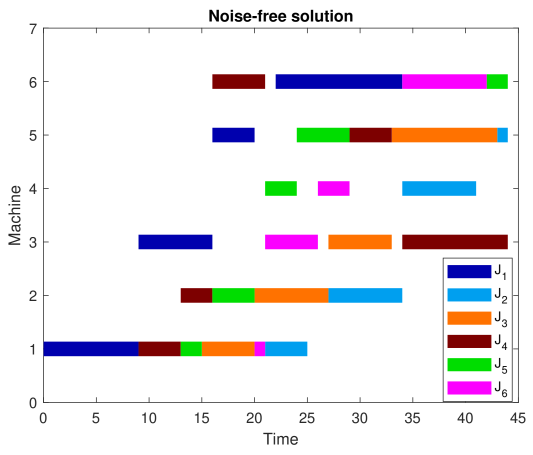
Table 3.7: Processing times

	M_1	M_2	M_3	M_4	M_5	M_6
J_1	9	5	7	10	4	12
J_2	4	7	3	7	1	10
J_3	5	7	6	3	10	1
J_4	4	3	10	6	4	5
J_5	2	4	7	3	5	2
J_6	1	6	5	3	6	8

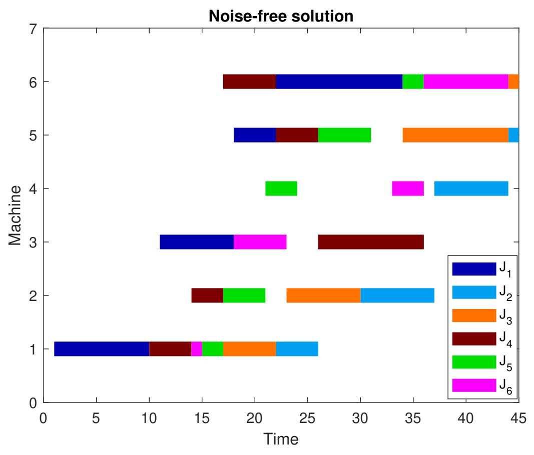
Table 3.8: Jobs' planned and real release

	R_0^{plan}	R_0^{real}
J_1	0	0
J_2	2	0
J_3	4	2
J_4	7	4
J_5	10	7
J_6	12	14

The planned release time for a job can deviate by a maximum of 3 units of time, either ahead of schedule or delayed. In contrast, the MPC's prediction horizon spans 2 units and inherently considers the scheduled release time due to the unknown actual release time. Moreover, it is assumed that jobs keep the scheduled order, avoiding possible swaps arising from concurrent delays and advancements of consecutive jobs.



(a) Offline, $\Omega = 0$



(b) MPC-based, $\Omega = 0$

Figure 3.8: Comparison between offline (a) and MPC-based (b) scheduling solutions in the case of deterministic processing times. Both solutions are comparable in terms of completion time.

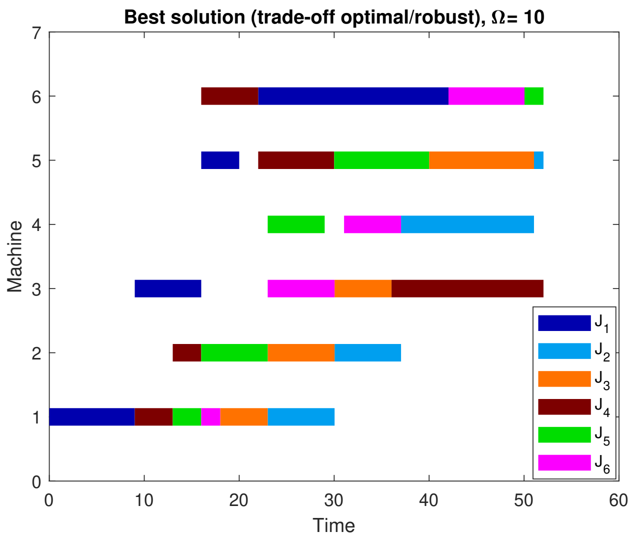
Fig. 3.8 illustrates the optimal solutions in the absence of disturbances in processing times. It can be observed that the completion time is nearly identical: 44 using the offline technique and 45 with the MPC-based approach. The key distinction lies in the fact that the latter does not necessitate deterministic knowledge of the entire system but rather dynamically adapts to the introduction of new jobs within the shop floor.

Table 3.9: MPC dynamic choice of the routing in the noise-free scenario

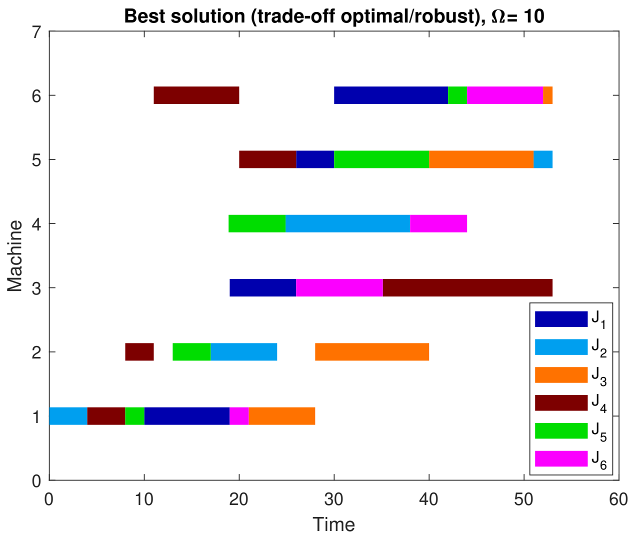
Jobs in the shop	Chosen alternatives
J_1	γ_2
J_1, J_2	γ_2, γ_4
J_1, J_2, J_3	$\gamma_2, \gamma_4, \gamma_6$
J_1, J_2, J_3, J_4	$\gamma_2, \gamma_4, \gamma_6, \gamma_8$
J_1, J_2, J_3, J_4, J_5	$\gamma_2, \gamma_4, \gamma_5, \gamma_8, \gamma_{10}$
J_1, J_2, J_3, J_4, J_5, J_6	$\gamma_2, \gamma_4, \gamma_5, \gamma_8, \gamma_{10}, \gamma_{13}$

This reasoning is highlighted in Table 3.9, which refers to the noise-free scenario and points out the chosen path at each arrival of a new job in the shop. The adaptability of the MPC-based scheduling becomes evident in the context of J_3. Specifically, it is initially scheduled according to the routing specified by γ_6. However, upon the subsequent arrival of J_5, J_3 is dynamically rescheduled using the routing strategy defined by γ_5. This dynamic adjust-

ment remains feasible because J_3 has not yet been allocated to a machine that would render altering its routing unviable, and it ensures the maintenance of competitive performance levels.
In order to shorten the computational time needed to solve the optimization problem, the alternatives listed in Table 3.6 are automatically eliminated once a job follows a divergent path incompatible with the continuation of said alternative. Similarly, Fig.3.9 depicts a comparison between solutions when processing times are perturbed, with a magnitude of $\Omega = 10$, while considering the worst-case distribution of these disturbances. In this scenario as well, the completion time remains nearly identical, despite the presence of disturbances.



(a) Offline, $\Omega = 10$



(b) MPC-based, $\Omega = 10$

Figure 3.9: Comparison between offline (a) and MPC-based (b) scheduling solutions in the presence of delayed processing times. Completion time is 52 in the ideal offline scenario, representing the minimum achievable value, and 53 in the dynamic scheduling, thus maintaining competitive performance.

Furthermore, from a computational standpoint, the MPC-based algorithm offers substantial benefits. Although there are more computations involved in managing data structures and implementing the controller, it is important to highlight that the initial scheduling only

involves a subset of the total jobs, resulting in faster optimization. As the number of jobs increases, on the other hand, much of the routing within the shopfloor for already present jobs has already been determined, thus minimizing its involvement in the optimization problem. Consequently, the pool of potential alternatives to be analyzed is significantly reduced. For these reasons, the implementation of the MPC-based algorithm results in an enhancement of computational performance. In the examined case study, it has been observed a significant 30% reduction in execution time, and it is reasonable to assume that a larger number of jobs corresponds to a more pronounced reduction in computational time, due to the algorithm's operational approach. Indeed, at each iteration, the MPC-based algorithm exclusively schedules jobs currently inside the shop floor as well as those anticipated to imminently arrive (i.e. within the prediction horizon). This selective scheduling enables the algorithm to effectively manage a limited number of jobs with each invocation, as opposed to offline scheduling.
To further substantiate the validity of the approach, Fig. 3.10 presents a robustness analysis conducted through 1000 simulations for the MPC-based scheduling, for each value of Ω. Processing times were subjected to random perturbations across the machinery, and a feasible solution with the same routing (i.e., identical γ values) was successfully determined. The graph proves that, across a wide range of scenarios, the solution provided by the approach guarantees better performance, representing the best trade-off between optimality and robustness for the given FJSP.

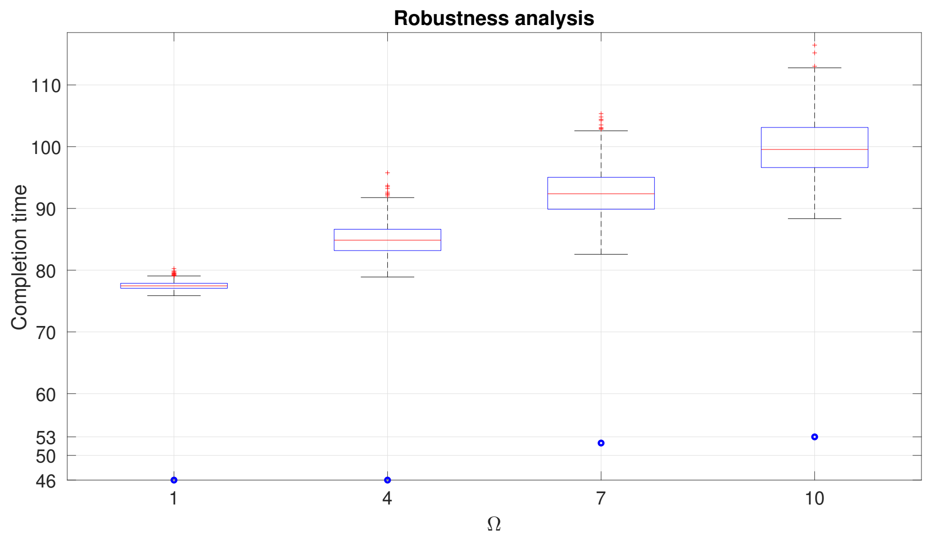


Figure 3.10: Robustness analysis of 1000 simulations for each magnitude of noise. Solutions present the same path, different feasible sequencing on machines, and randomly generated disturbances. Blue points represent the solutions obtained with the proposed algorithm in the case of worst delays

Chapter 4

Platoon control in manufacturing

In the previous chapters, platooning control techniques and manufacturing scheduling were thoroughly explored, elucidating their intricacies. This chapter marks the transition into an interdisciplinary realm where the boundaries between the two fields of research are merged. Here, the focus shifts towards the integration of platooning-based control techniques into smart manufacturing systems. In this chapter, the potential benefits of such integration are examined. By merging these domains, the aim is to enhance manufacturing efficiency. Through the analysis of case studies and empirical data, insights into the optimization of material flow, reduction of makespan, and enhancement in general productivity within manufacturing environments are revealed. This chapter is centered on the exploration of possibilities and practical implementations of platooning in manufacturing. The objective is to uncover how this innovative approach can transform manufacturing processes into more agile, responsive, and efficient operations.
In the following, the utilization of AGVs as integral components of platooning formations has been treated. These AGVs, equipped with advanced autonomous navigation systems, collaborate within platoons to streamline material handling, reduce bottlenecks, and enhance overall system efficiency.

4.1 AGV control: Potential field

The trajectory planning of AGVs has been accomplished using the potential field controller. This method identifies the forces of attraction and repulsion between objects within the system. In the manufacturing case study, the destination machine acts as the point of attraction, while other AGVs, machines, and recharging stations are points to avoid. This is necessary because trajectory planning involves designing paths that do not intersect with other elements or shared resources in the system.

The general equation of the resulting potential field is:

$$\vec{P}_{tot} = \vec{P}_{att} + \sum \vec{P}_{rep} = \vec{\nabla} V(\underline{q}) \tag{4.1}$$

$$\begin{cases} \vec{P}_{att} = K_a(\underline{q} - \underline{q}_g) \\ \vec{P}_{rep}(\underline{q}_c) = \begin{cases} K_r(\frac{1}{d(\underline{q})} - \frac{1}{d_0})\frac{1}{d^2(\underline{q})}\frac{\underline{q}-\underline{q}_c}{\|\underline{q}-\underline{q}_c\|} & \text{if } d(\underline{q}) < d_0 \\ 0 \quad \text{if } d(\underline{q}) \geq d_0 \end{cases} \end{cases} \tag{4.2}$$

where $\underline{q}$ represents the current pose of the agent, $\underline{q}_g$ its goal position, d_0 is the maximum distance to consider an interaction in the potential field (i.e., the repulsion is zero when the distance between two objects is greater than d_0), $\underline{q}_c$ is the current pose of the obstacle, K_a and K_r respectively the attraction and repulsion gains.
The computation of $\vec{P}_{rep}(\underline{q}_c)$ has to be repeated for each obstacle in the system. The position of static obstacles is fixed and does not change over time, while for dynamic obstacles (i.e., other AGVs) the repulsive force is computed with respect to their instantaneous position. To differentiate between the two types of obstacles, different repulsive gains can be designed, namely K_{rs} for static obstacles and K_{rd} for dynamic ones.
The potential field has to be decomposed into x and y components in order to retrieve the corresponding value of speed to supply to AGVs. To this end, it must be noted that the force gradient is:

$$\vec{\nabla} V(\underline{q}) = \begin{bmatrix} \frac{\partial V}{\partial x} & \frac{\partial V}{\partial y} \end{bmatrix}^T = [F_x F_y]^T \tag{4.3}$$

by incorporating the mass of the AGV, whether with or without payload, the corresponding velocities along the x and y axes can be determined.
As the last thing, to detect and design a trajectory that takes into account physical borders of the SMS, the repulsive force needs to be extended with an additional term to consider borders:

$$\vec{P}_{rep\_b} = \sum_{j=1}^{J} \left(-K_{r\_b} \left(\frac{1}{|\vec{r_j} - \vec{q}|} - \frac{1}{d} \right) \left(\frac{\vec{r_j} - \vec{q}}{|\vec{r} - \vec{q}|^3} \right) \right) \tag{4.4}$$

where K_{r_b} is the gain for the repulsive force from the borders, r_j is a point on the border, and J is the total number of points considered on the borders. Indeed, to account for the borders effectively, a discretization process is employed, wherein a finite number of points are selected to represent the boundaries.

4.1.1 Emergency controller

To ensure safety, at each iteration vehicles have to compute the distance with respect to neighbors in order to verify that a minimum safety distance is maintained with other

elements in the system.
In a centralized fashion, a single authority would be in charge of this check for each AGV. This would imply comparing every pair of vehicles' position and would result in a too computationally heavy task. For this reason, it is logical to assume that each AGV is provided with its own sensors to perform the assignment in a decentralized way.
In the event that the prescribed safety distance is disregarded, a precautionary low-level protocol is triggered to prevent collisions. Within this particular scenario, one of the two vehicles comes to a halt, while the other vehicle, prioritized because of its shorter due date (or its release order in case of an equivalent due date), is permitted to proceed toward its destination. The former vehicle resumes its motion once the latter has safely exited the emergency radius.

4.2 Platooning AGVs

AGVs can be merged in platoons by aligning with various criteria, each of which holds the potential to enhance manufacturing operations. One fundamental criterion for AGV platooning is the utilization of shared pathways. This involves aligning vehicles that travel along similar routes into cohesive formations, a strategy aimed at minimizing traffic congestion and optimizing resource allocation within the manufacturing facility. This approach is particularly valuable in shop floors equipped with railways, where vehicles lack the freedom to move freely throughout the facility. Another criterion for platooning is a common destination or shared goals. AGVs that are all headed toward the same resource, whether it be a specific machine, assembly station, or storage area, can collaboratively navigate the shop floor. Additionally, AGVs can align within platoons when they share a common objective during their shop floor operations. For instance, they may collectively aim to complete the products they are carrying as rapidly as possible. This objective-driven platooning strategy optimizes task allocation, as well as AGVs working together to conserve energy, especially in cases where vehicles have low battery levels. By grouping together and strategically coordinating their movements, AGVs can minimize energy consumption and prolong their operational lifespan, ensuring uninterrupted manufacturing processes.

4.2.1 Reactive platoon-based controller for AGVs

The reactive platooning-based procedure serves as a coordination method for AGVs, designed to prevent shopfloor congestion and align their path trajectories based on priority criteria. This procedure dynamically forms a subset of AGVs with a temporary shared objective, distinct from the individual objectives of the group members. Specifically, it operates as follows:

- The platoon is dynamically created during execution, comprising two types of AGVs:
  - The platoon leader, possessing the highest priority.
  - AGVs that partially or fully overlap with the path of the platoon leader.
- Subsequently, the speeds of the second type of AGVs are adjusted to minimize congestion for the platoon leader.
- At each iteration, all AGVs revert to their initial speeds, and the process repeats for AGVs still operating within the shop floor.

This dynamic procedure, explained in Algorithm 2, continuously adds or removes AGVs from the platoon, depending on the production path of the platoon leader.

Algorithm 2 Platoon-based cooperation procedure

```
1: while $\#AGVs \geq 2$ do
2:     define AGV with (max priority) as platoon leader
3:     for all AGVs do
4:         if AGV $i^{th}$ shares routing path with leader then
5:             add AGV $i^{th}$ to platoon
6:             if AGV $i^{th}$ is in the shared path then
7:                 $\text{speed}^i = \text{speed}^i + \text{speed-factor}$
8:             end if
9:         end if
10:    end for
11: end while
```

The speed factor denotes an increase in the velocity of shuttles and has been empirically determined to be 10% to 20% faster than the regime speed until the next iteration with the control architecture. This approach incorporates the advantages of platooning by grouping a subset of shuttles under a common cooperation rule. The key contribution of this approach lies in its dynamic and iterative policy, which adjusts shuttle speeds in real-time to cooperatively assist in achieving the objectives of a prioritized shuttle. As a result, production progress is sequenced in accordance with current manufacturing requirements, prioritizing shuttles based on their assigned priority.

4.2.2 Platoon-based control with time of arrival optimization criterion

Another platoon-based control approach can intervene to group and manage the displacement of AGVs headed to the same shared resource, aiming to provide the optimal speed profile that minimizes energy consumption and ensures that AGVs reach their destination precisely when it becomes available. If the target machine is currently in use, AGVs are required to wait until it becomes accessible. Therefore, optimizing AGV speeds to synchronize their arrival times with machine availability is a strategic approach to simultaneously minimize waiting times and energy consumption. The platoon-based approach, implemented to achieve precise coordination between AGVs and machine availability, results in distinct behaviors for the AGV closest to the shared resource, namely the platoon leader, and the followers. The leader adheres to its control law provided by the potential field, as discussed in Section 4.1, while the movement of each follower (denoted as the i^{th} AGV) is regulated based on the arrival time at the shared resource of the preceding AGV in the platoon formation, and the processing time of that resource:

$$V_i = \frac{\sqrt{(x_i - x_m)^2 + (y_i - y_m)^2}}{t^a_{i-1} + p_m} \tag{4.5}$$

where (x_i, y_i) represents the position of the i^{th} AGV, (x_m, y_m) the position of the shared machine, p_m its processing time, and t^a_{i-1} stands for the arrival time of the preceding AGV. To maintain consistency, the arrival time of the leader is computed under the reasonable assumption of constant speed throughout the entire path from its initial position to its destination.

4.3 Platoon-based control in AIP-PRIMECA

In order to validate a first, simple platoon-based technique on a manufacturing system, the reactive control of Section 4.2.1 is benchmarked on the AIP-PRIMECA production cell explained in Section 3.3, suitable to determine the feasibility and potential benefits of incorporating a reactive control procedure into the material handling system to improve production performance.

The benchmark for the flexible manufacturing system (FMS) consisted of seven machines capable of processing seven types of products, all transported using a shuttle-based material handling system. A virtual testbed model of the selected case study has been created, aiming to replicate the distributed handling control system and comprising two key layers: the local and physical layers. The local layer, programmed in Python using the MESA framework, represented the manufacturing system in a virtual environment, wherein the platoon-based control procedure was implemented. The physical layer has been developed using an agent-based simulation software, Netlogo, to simulate the physical world and production execution. A graphical representation is illustrated in Fig. 4.1. To establish a connection between the two layers, the pyNetlogo library was utilized. This configuration enabled continuous communication between the AGVs in the Python program and their counterparts simulated in Netlogo, effectively creating a digital twin within the Python program. To facilitate comparative analysis, a second instance of the testbed has been created to assess the proposed platoon-based approach against a no-control policy.

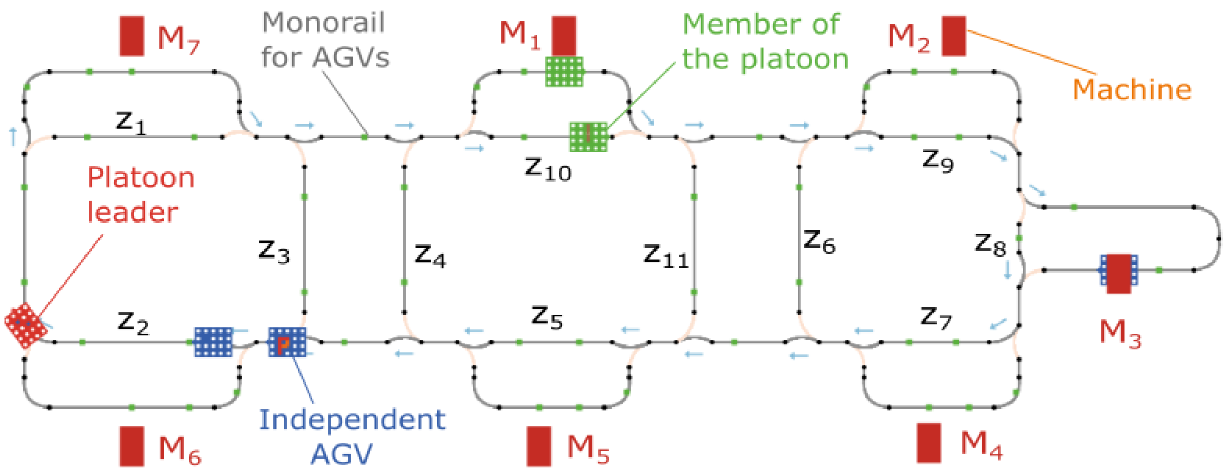


Figure 4.1: FMS representation: the red agent is the platoon leader, the green vehicles are part of the platoon and the blue vehicles are independent agents. z_i denotes the i^{th} monorail zone, used for result analysis

Then, a set of production orders has been created, changing the number of products and their variety, to obtain a diverse degree of complexity. The experimental protocol has been tested on 10 instances, whose complexity has been computed with the formula:

$$f(n, t) = n + 3 \cdot t \tag{4.6}$$

where n is the number of products, and t is the number of product types. The weighted load is a simple parameter included in the experiment in order to preliminarily identify the effect of product and order diversity within the platoon-based procedure.

Table 4.1: Experimental instances and weighted load (complexity)

Scenario	Number of AGVs (n)	Number of product types (t)	Weighted load
Instance 1	5	1	8
Instance 2	7	1	10
Instance 3	5	2	11
Instance 4	7	2	14
Instance 5	5	4	17
Instance 6	7	4	19
Instance 7	10	4	22
Instance 8	7	7	28
Instance 9	12	6	30
Instance 10	15	7	36

Table 4.1 provides an overview of the tested instances, while Fig. 4.2 presents the experimental results, demonstrating a general increase in complexity as the weighted load increases, with few exceptions.This increase in complexity is primarily driven by the number of products rather than their variety. While a variety of products introduces greater uncertainties in their routing within the system, it is the number of products that creates congestion, especially during the loading and unloading phases.

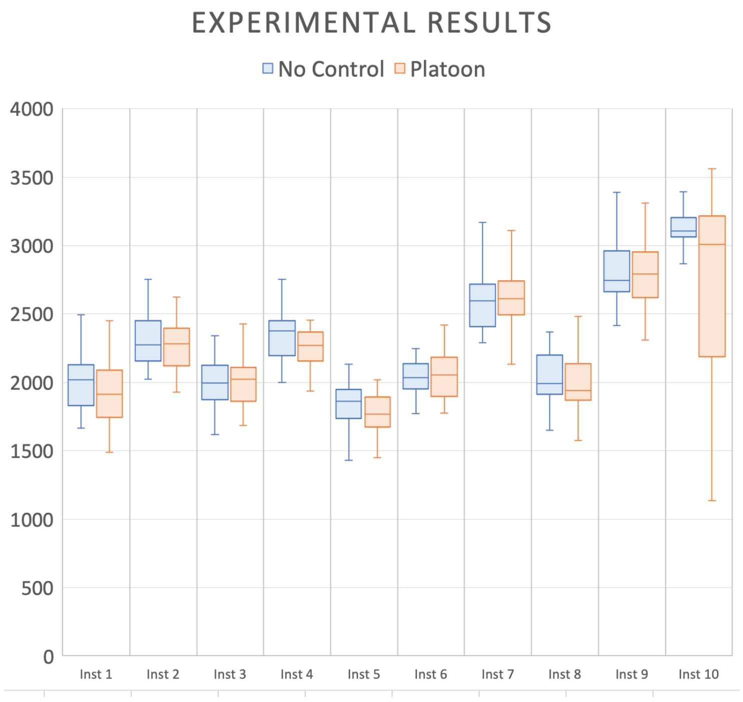


Figure 4.2: Comparison of control procedures in makespan distribution in seconds for 10 instances with increasing complexity

To assess the impact of platoon-based control compared to a no-control policy, two key response variables have been used as indicators: the makespan (representing task completion time) and the congestion factor throughout the FMS. The FMS has been divided into 11 zones, each linked to a specific monorail in proximity to the workstations.
A non-parametric ANOVA-type statistic, specifically a MANOVA analysis [114], was utilized to assess the significance of the control policy procedure factor for each response variable independently. This multivariate analysis of variance aimed to evaluate the difference between the Platoon and No-Control approaches as a single independent variable across multiple outcome variables. While Fig. 4.2 showed similar average behavior between the approaches, the MANOVA was conducted to determine if there were significant variability differences.

Table 4.2: Non-parametric MANOVA

	Df	SS	R2	F	Pr($> F$)
Instance_factor	9	11.735	0.691	149.902	0.001
AGV_factor	1	0.053	0.003	6.078	0.004
Instance_factor: AGV_factor	9	0.160	0.009	2.042	0.016
Residual	580	5.045	0.297		
Total	599	16.993	1.000		

Table 4.2 displays the standard MANOVA results, including the relationships between instances, AGV speed, and their interactions, with respect to response variables such as makespan and zone occupancy.
A p-value is a statistical measure used to assess the significance of observed results in hypothesis testing. It quantifies the likelihood of obtaining results as extreme as those observed, assuming the null hypothesis is true. In this study, a significance level of 0.05 was chosen. When the p-value is less than 0.05, it indicates that the observed results are unlikely to have occurred by random chance alone. According to results in Table 4.3, variables with p-values below this threshold, such as makespan and occupancy in zone 10, are considered significant. An additional non-parametric analysis was performed for Makespan and Occupancy Zone 10 since the assumptions for traditional MANOVA were not met. The results, presented in Table 4.4, indicate that the intervals are narrower for makespan and occupancy of zone 10, demonstrating better performance in these response variables.

Table 4.3: p-values for AGV control

Response variable	p-value for AGV velocity control	Response variable	p-value for AGV velocity control
Makespan	0.0006	Zone6	0.1867
Zone1	0.8994	Zone7	0.3290
Zone2	0.5554	Zone8	0.0540
Zone3	0.8062	Zone9	0.2006
Zone4	0.4674	Zone10	0.0065
Zone5	0.4165	Zone11	0.8955

Table 4.4: Confidence intervals for ANOVA-Type Statistic Ranking of AGV Velocity Control for Zone 10 and Makespan

Response variable	Policy	Rel. effect	Std. error	Lower value	Upper value
Makespan	No control	0.5234	0.0068	0.5101	0.5366
	Platoon	0.4766	0.0068	0.4634	0.4899
Zone 10	No control	0.5142	0.0052	0.5039	0.5244
	Platoon	0.4858	0.0052	0.4756	0.4961

In general, the results suggest a significant effect of the control policy, as evident in the makespan and zone 10. The confidence intervals indicate that platoon-based control outperforms the no-control policy, as the ranked values are consistently lower. This outcome is reasonable because increasing the speed of certain agents accelerates the processing of products. Even though it may lead to congestion in some parts of the system, this did not occur in the analyzed case study. Moreover, congestion is reduced, especially in zone 10, which represents the critical bottleneck in the system, given its close connection to the loading/unloading workstation. In conclusion, it can be stated that the reactive platoon-based control policy represents a valuable initial option for platoon-based control in manufacturing.

4.4 Optimizing AGV control with platoon-based techniques and recharging station scheduling in SMS

The final work of this thesis considers a comprehensive case study in SMS, wherein the objective is to minimize the makespan by employing various control techniques based on the AGVs' states over time. AGVs are modeled using the first-order kinematic model of Section 2.1.1, which ensures a fast calculation of the system's evolution. To enhance the realism and alignment with AGVs in manufacturing systems, saturators for speed and rate of speed (i.e. acceleration) are incorporated.
Effective trajectory planning is crucial to ensure rapid and collision-free routing within the trackless manufacturing system, providing optimal speed profiles in terms of efficiency and energy consumption. The term 'trackless' refers to the absence of fixed physical rails on the factory floor, which enables AGVs to operate autonomously and navigate through the manufacturing system using alternative guidance mechanisms, such as sensor-based technologies. Additionally, given the AGVs' susceptibility to battery discharge, implementing a recharge scheduling plan is fundamental to ensure uninterrupted production while minimizing waiting times at the recharging stations. For simplicity, we assume that the decision

on recharge is made only between the completion of one product and the potential start of a new one. Thus, the control structure designed in this work can be divided into three parts:

- Trajectory planning and control for independent AGVs: it focuses on AGVs that are going to machines. It includes the following steps:
  - Compute the significant neighborhood by identifying vehicles within the attraction radius
  - If there are vehicles within the safety radius, perform an emergency maneuver to avoid imminent collision with the principles of Section 4.1.1
  - Apply the potential field controller of Section 4.1 to compute the velocity for the AGVs

  This technique is distributed, as each agent computes its own potential field based on interactions with neighboring agents and its local perception of the environment, which may even be enhanced in a real manufacturing by sensors placed on AGVs. Thus, there is no central authority.

- Recharge convenience check for AGVs that ended a product process: it involves AGVs that are going to the queue after completing a product process. A decision-making process is applied to verify whether it is convenient or not to proceed with a recharge

- Platoon control for AGVs going to shared resources: it focuses on AGVs that are heading towards recharging stations or unloading units, grouping them into platoons and controlling each platoon with the specialized strategy of Section 4.2.2

By dividing the algorithm into these three parts, the control structure addresses different aspects of AGV control, including trajectory planning, safety considerations, recharging decisions, and platoon control. Each part handles a specific scenario and contributes to the overall control and coordination of the AGVs within the system. Algorithm 3 represents a schematic of the abovementioned approach.

Algorithm 3 Complete control algorithm

```
Require: Positions and destinations of AGVs
Ensure: Speed profile for AGVs
 1: procedure ControlAlgorithm
                                                  ▷ a) AGVs going to machines
 2:     for AGVs going to machines do
 3:         Identify significant neighborhood     ▷ Vehicles under the attraction radius
 4:         if neighbor within safety radius then
 5:             Perform emergency maneuvers
 6:         else
 7:             Apply the potential field controller and supply velocity to AGV
 8:         end if
 9:     end for
                                                  ▷ b) AGVs going to the queue
10:     for AGVs going to the queue do
11:         if recharging is convenient then
12:             AGV.destination = recharging station
13:         end if
14:     end for
                                                  ▷ c) AGVs going to unloading unit
15:     for AGVs going to unloading unit do
16:         UnloadingPlatoon.append(AGV)
17:     end for
18:     Apply control for UnloadingPlatoon
                                                  ▷ d) AGVs going to recharging stations
19:     for AGVs going to recharging stations do
20:         RechargingPlatoon.append(AGV)
21:     end for
22:     Apply control for RechargingPlatoon
23: end procedure
```

4.4.1 Battery recharging reactive decision-making

AGVs are equipped with batteries that adhere to standard charge/discharge cycles. Realistic battery values have been sourced from Safelog AGVX1 models [115] and then scaled to expedite the discharging rate for simulation purposes. Specifically, the original discharging rate has been divided by 20 to prevent the simulation from being excessively time-consuming.

The charging time is expressed as:

$$t = k_1 e^{k_2 x} \tag{4.7}$$

where k_1, k_2 are two constants used to model the battery capacity of charge and x is the depth of discharge (DoD) representing, in percentage, how much the battery is discharged. In other words, the state of charge (SoC) is $SoC = 100 - 100DoD$. To model the charging operations with a similar time of the AGVX1 model (for the one-battery model), the following values of k_1, k_2 have been chosen:

$$k_1 = 121, \qquad k_2 = 2.7 \tag{4.8}$$

These values have been taken following the reasoning of [116, 117] and adapted to the given datasheet.
On the opposite, the discharging overtime has been designed taking into account different discharge ratios based on the AGV's speed and whether or not it carries a payload:

$$SoC = SoC_0 - C_1 V \Delta t - C_2 P \Delta t \tag{4.9}$$

where SoC_0 is the state of charge at the previous instant, Δt is the time elapsed between the two sampling instants, V is the speed and P a boolean to state if the AGV is carrying a payload. The speed is assumed to remain constant throughout the time interval ΔT. Constants C_1 and C_2 have been modeled to align with the information provided in the datasheet. In the absence of a payload, the battery has a 4-hour lifespan, which corresponds to $C_1 = 0.004$. For C_2 empirical considerations have been done: in particular, it is reasonable to assume that a payload decreases the battery capacity of an additional term estimated as 25% more (i.e. $C_2 = 0.5 \cdot C_1$), although further studies should be done on the influence of payload weight in battery discharging. Thus, the equation of the battery discharging is:

$$SoC = SoC_0 - 0.004\Delta t(V + 0.5P) \tag{4.10}$$

This model operates under the assumption that when an AGV is stationary, there is no battery discharge. After completing one product process, AGVs have two options: go to the waiting queue to start another product process or to the recharging station to recharge batteries. It is assumed that the battery is fully recharged once in the recharging station and the process cannot be interrupted.
To choose whether charging is convenient or not, four different terms help in building the decision:

- the current SoC
- the number of batteries in the AGV
- the number of AGVs in the waiting queue

- the number of free recharging stations

Borderline values of SoC are treated straightforwardly, i.e. recharging is not allowed if $SoC \geq 80\%$ and it is mandatory if $SoC \leq 20\%$, which represents the minimum safety SoC for AGVs to complete a cycle of one product process. If SoC reaches this level, AGVs are compelled to enter a power-saving mode with limited speed.
On the contrary, for values within the range of 20% to 80%, the following equation has been designed to provide a convenience coefficient (CC) for recharging:

$$CC = K_1 \frac{R_{free}}{R_{tot}} + K_2 \frac{AGV_{wait}}{AGV_{tot}} + \frac{K_3}{B} \frac{SoC}{100} \tag{4.11}$$

subject to:

$$K_1 + K_2 + K_3 = 1 \tag{4.12}$$

where R_{free} is the number of available recharging stations, R_{tot} the number of total recharging stations, AGV_{wait} the number of AGVs waiting in the queue to move to the loading unit, AGV_{tot} the total number of AGVs, B the number of batteries equipped on the AGV, and K_1, K_2, K_3 are gains used to prioritize respectively the number of free recharging stations, the number of AGV waiting in the initial queue and the current state of charge of the AGV. Eq. 4.11 and 4.12 provide a value between 0 and 1 to choose whether the AGV can profit from a recharge or not. CC values above 0.5 indicate an advantage in recharging, while lower values suggest the AGV postponing the recharge and continuing with the routing of products.

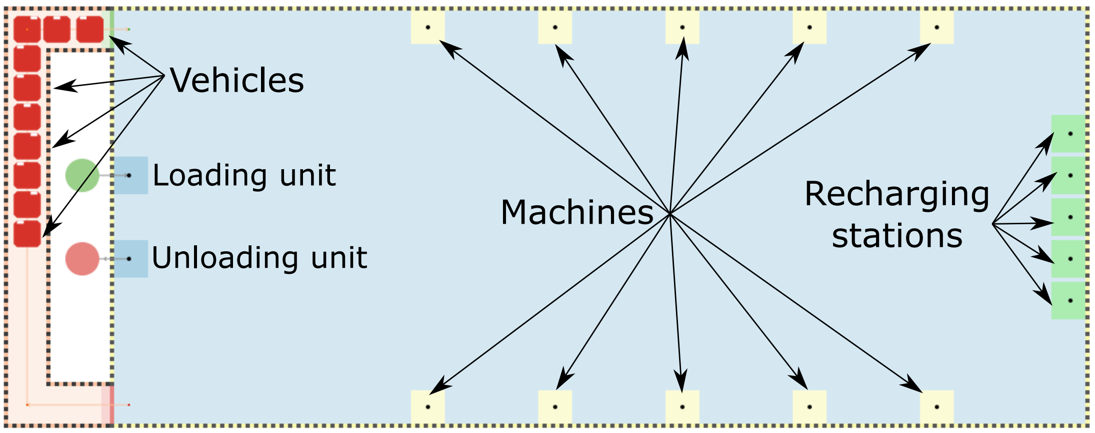


Figure 4.3: Representation of the manufacturing system at the beginning of the simulation. Products are mounted on top of vehicles within the loading unit

4.4.2 Analysis of a complete control system for production within manufacturing production system

The case study examines a rectangular production cell consisting of 10 machines, one loading unit, one unloading unit, and five recharging stations, within a shop floor measuring 23 × 11 m. Ten AGVs, four of which are equipped with one battery (AGVs of Type 1), three with two batteries (Type 2), and the remaining three with three batteries (Type 3), traverse the shop floor. Figure 4.3 illustrates the manufacturing system implemented in NetLogo. A buffer to store vehicles during the waiting phase is required, but remains out of control scope. Moreover, due to the symmetry of the system, vehicles may have crossing trajectories in the default scenario (i.e., moving straightforwardly to the destination); thus, a trajectory planning controller seems fundamental to improve performance.

Type	**Weight**	**Operations**
1	20	1-4-7
2	20	1-5-8
3	20	1-3-9
4	20	1-4-6-8
5	20	2-5-8-9
6	30	1-5-7
7	30	2-5-8
8	30	3-5-9
9	30	3-4-6-10
10	30	3-5-7-10

Table 4.5: List of products

In the investigated manufacturing system, the routing of products is fixed due to the lack of machines' flexibility.

Scenario #1 4 product types	**Scenario #2** 6 product types	**Scenario #3** 8 product types
1	1	1
3	2	2
5	1	10
3	4	3
3	4	6
1	5	5
1	8	10
7	10	7
5	2	9
5	8	9
7	8	1
5	5	6

Table 4.6: Production order, repeated 3 times (36 products in total) for each simulation

Table 4.5 presents comprehensive information about the types of products, while Table 4.6 shows the different production orders used for simulation, repeated 3 times to guarantee a proper discharge of batteries and test the algorithm in medium-term scheduling with the battery recharging, since three distinct types of AGVs are considered, each equipped with a variable number of battery slots. Understanding these fixed routes and operations is essential for optimizing AGV trajectories to ensure efficient and reliable movement of products throughout the system. In fact, as there are no railways, AGVs are free to move within the shop floor, and thus their trajectory needs to be optimized to avoid collisions and minimize congestion. When carrying a product, their maximum speed is $0.5m/s$, and $0.8m/s$ otherwise. A speed rate limiter of $0.1m/s$ has been inserted to avoid abrupt changes in their speed that might make the product fall off the AGV.
Algorithm 3 is iteratively executed every 10 iterations, corresponding to 0.5 seconds in the real scenario. This sampling time enhances responsiveness by allowing the control system to gather updated information about the AGV's state and environment more frequently and keeping a balance between computational efficiency and control accuracy. Moreover, it ensures that the control algorithms can be executed within a reasonable time frame without compromising system performance or introducing significant delays. Parameters of the control algorithm used for simulations are represented in Table 4.7, even if their tuning can drastically upset system's performance and represent thus a crucial part of the process. The processing time of operational machines is fixed to 30 seconds, loading is immediate and unloading requires 15 seconds.

Table 4.7: Parameters of the control algorithm

K_a	K_{rd}	K_{rs}	d_0 [m]	d_{min} [m]	K_1	K_2	K_3
20	350	300	10	0.8	0.25	0.5	0.25

Multiple simulations were conducted for each scenario, and the results in Fig. 4.4 confirm that higher product variety leads to increased complexity in routing and in the generation of safe trajectories, which needs to ensure collision avoidance for the whole production. As a consequence, this heightened complexity generally leads to longer makespan and necessitates AGVs to undergo more frequent recharges throughout the simulation.

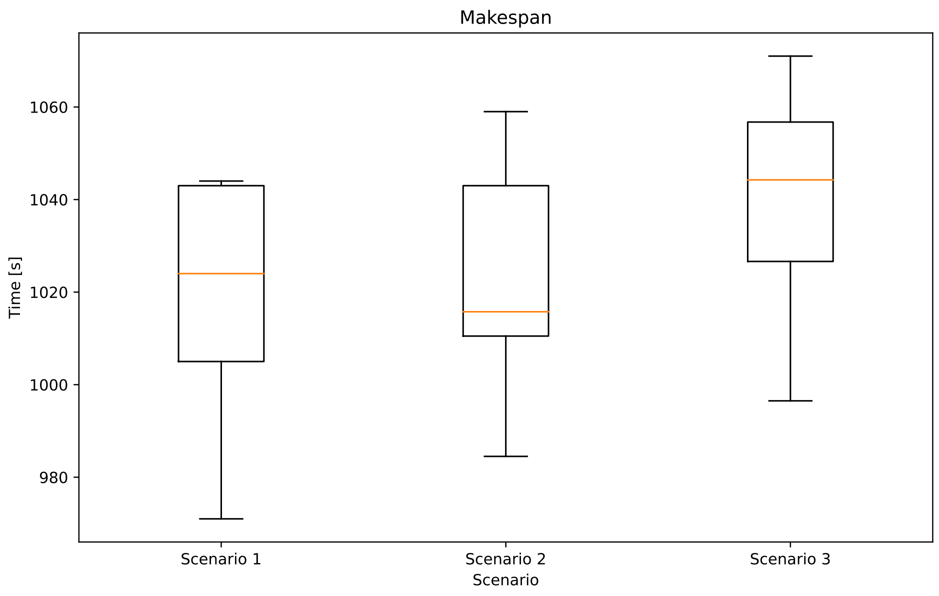


Figure 4.4: Makespan distribution over 10 simulations for each scenario

However, the simulations reveal the effectiveness of the proposed approach. The AGVs successfully complete production tasks without collisions, and their speeds consistently approach the maximum allowable, demonstrating that the potential field controller dynamically generates safe trajectories even at high speeds, ensuring performance competitiveness. This is illustrated in Fig. 4.5, which shows the speed distribution of four AGVs in the three different scenarios.

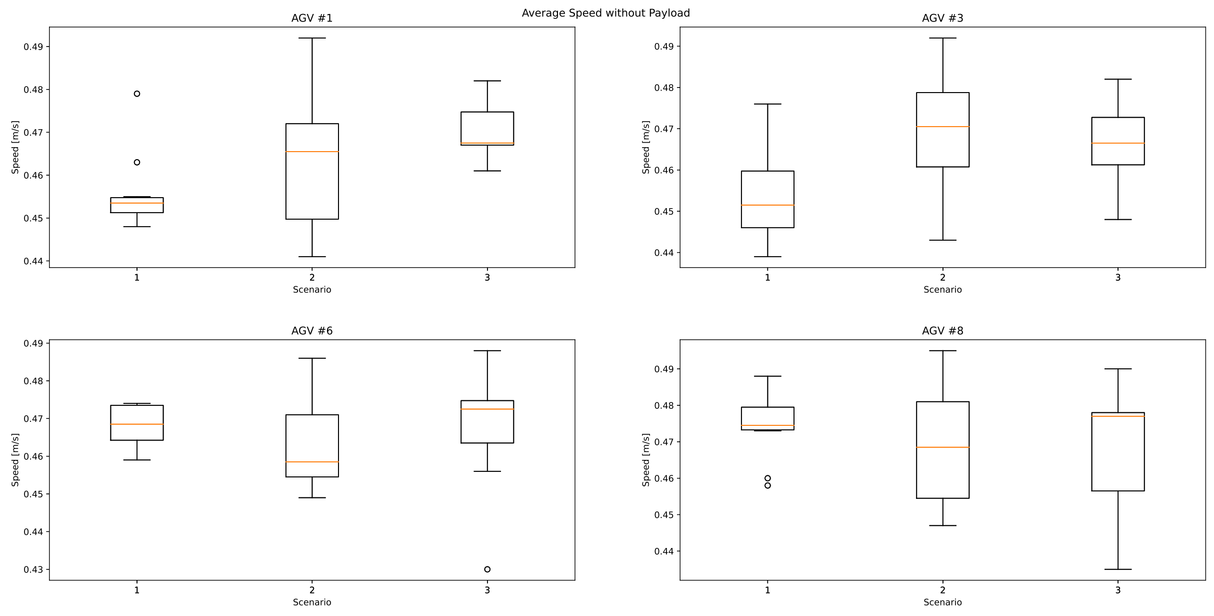


Figure 4.5: AGVs average speed distribution of four AGVs across the three scenarios

Furthermore, recharging scheduling is accomplished as AGVs autonomously select the optimal recharging times based on the current system state and their remaining battery capacity. Across all simulations, there were consistently at least two available free recharging stations, suggesting that out of the five initially available, the number can potentially be reduced to three without affecting system performance.
Notably, it is observed that primarily AGVs of type 1 required recharging in the given scenarios. Approximately 80% of AGVs that underwent recharging were of type 1, with the remaining 20% being of type 2. AGVs of type 3, in a medium-term production chain, did not require recharging. This observation aligns with the data shown in Fig. 4.6, where the number of AGVs requiring recharging fluctuates between 5 and 8 across all simulations. Specifically, AGVs of type 1 consistently necessitate at least one recharge throughout the entire production process, while the decision for AGVs of type 2 to recharge depends on the system's state.

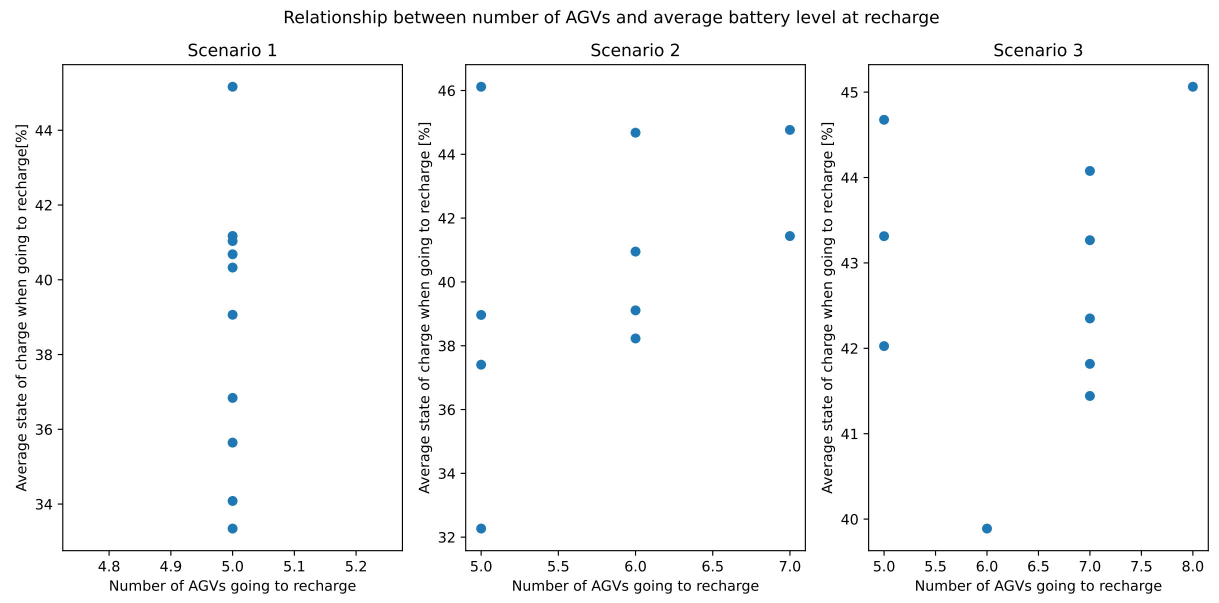


Figure 4.6: Distribution of the average battery level when going to recharge stations over 10 simulations for each scenario

The figure also illustrates that, on average, AGVs head to recharging stations with their remaining battery levels consistently below half of their total capacity. However, there is still adequate power available before reaching the power-saving mode threshold ($< 20\%$). This finding instills confidence in the robustness of the reactive decision-making control policy, even in scenarios or situations where unforeseen events may cause AGVs to delay their recharging.
In conclusion, the analysis of this case study illustrates that the proposed approach empowers AGVs to operate consistently within the SMS, completing production processes without interruptions. This is achieved through trajectory planning and collision avoidance handled respectively by the potential field and emergency controllers, optimized machine utilization via platooning for shared resources, minimized energy consumption, and efficient recharging scheduling guided by the reactive decision-making process. These factors collectively contribute to uninterrupted and highly efficient production operations.

Chapter 5

Conclusion and future work

This thesis represents a comprehensive exploration of two research topics, autonomous vehicle platooning and smart manufacturing systems, that may appear unrelated at first glance, but ultimately converge on the broader theme of optimizing systems for efficiency and sustainability.

Initially, the platooning of multiple autonomous vehicles was explored, demonstrating that through the strategic application of advanced control techniques, such as robust team decision theory, LQT, and PID, these independent vehicles can act as a cohesive unit, achieving not only individual benefits but also contributing to traffic improvements. The studies presented in Sections 2.3 and 2.4 focus on platooning maneuvers designed specifically for HDVs. These maneuvers align with the platoon size discussed in [58], which aims to enhance transportation capacity and reduce fuel consumption, as demonstrated in [54]. The primary objective in both works is to achieve these benefits while minimizing alterations to acceleration patterns. The control techniques employed in both studies are drawn from well-established literature. For instance, [59] first introduced the concept of using LQ control for platoon formation in a string. Even if [61] has pointed out challenges at high speeds, applying LQR or LQT for HDVs ensures safety and maneuver accuracy given that vehicles of this type typically do not reach high speeds. These control strategies exploit straightforward yet realistic models for trucks, instilling confidence in their applicability in real-world testing environments. In these works, lateral displacement is not considered, but it may be integrated with future research to provide a more comprehensive approach to vehicle displacement control. The procedure outlined in [37] is a potential method for integration, employing Vehicle-To-Vehicle communication to enhance reliability.

Furthermore, platoon control was also examined within a macroscopic traffic environment to preliminarily demonstrate how one or more platoons can influence the throughput of an entire road portion. Platoons have a significant impact on traffic and in case of congestion they represent the main bottleneck due to their size and influence on throughput, as highlighted in [42, 43]. In the study discussed in Section 2.5, a dual-level controller was designed, drawing inspiration from [31]. The main difference lies in the retrieval of the desired speed, determined from the traffic model. A dedicated low-level controller for platooning, similar to the approach in [50], is employed to reach the target speed. Future research could explore the integration of the dual-level controller with platoon-actuated control, as proposed in [44], to anticipate and potentially prevent congestion during bottleneck scenarios. Additional enhancements in platoon formation within traffic could be achieved through predictive control methods, such as those discussed in [18], in order to take into account traffic conditions to decide when to form platoons, particularly at critical points like on-ramps.

In addition, implementing an event-based controller on a WMR as a practical example in Section 2.6 reinforces the versatility and adaptability of the research, demonstrating how control techniques can be deployed on simpler, cost-effective hardware, that remains sufficiently reliable for an initial testing phase. In this context, hardware enhancements can be explored. This may involve incorporating more reliable sensors, even if it leads to slightly increased costs, and employing methods to enhance their ability to reconstruct the surrounding environment. For instance, equipment described in [39] can be applied to provide WMRs with indoor localization. Similarly, localization tags are under study for WMR navigation, with the goal of replacing line-tracking sensors with information derived from these tags, effectively creating an emulated GPS system and eliminating the need to draw roads for testing vehicle behavior. Additionally, achieving WMR platooning through Bluetooth or Wi-Fi communication can facilitate information exchange, enhancing the reliability of each vehicle within the environment and enabling the analysis of more complex scenarios.

In the second part of the thesis, product scheduling in both offline and real-time scenarios within manufacturing systems are treated, in order to optimize performance indexes such as makespan or minimize maximum regret. Offline scheduling algorithms were benchmarked against a model inspired by a real production cell in Valenciennes, France, to enhance the realism of the experiments. The applications of offline approaches in Sections 3.3.1 and 3.3.2 offer an innovative solution for addressing the FJSP with respect to the work in [78]. Formulating the problem as a flow-shop allows the utilization of various scheduling algorithms via MILP. Additionally, a newly designed iterative algorithm ensures competitive performance, particularly when uncertainties exist within the system. While these offline approaches may not be directly comparable to dynamic scheduling techniques, they effectively account for system uncertainties. They are particularly well-suited for scenarios involving small to medium production orders, given the inherent limitations of offline techniques, especially in manufacturing systems where production is highly predictable and less susceptible to unforeseen events, primarily influenced by delays in processing times.

Real-time scheduling offers a significant improvement in adaptability within FMS. It allows for dynamic production reorganization in response to the latest events occurring in the system. In contrast to other approaches found in the literature, such as those discussed in [83] and [84], the method outlined in Section 3.4 is based on MPC, a technique commonly employed in autonomous vehicles. In a custom case study, this approach demonstrated its potential and was compared to previous offline scheduling methods. It exhibited faster computational times and greater scalability, especially for larger production orders.
Future research, both in offline and online scheduling, may consider machine failures, as these situations may require products to be re-routed or temporarily held until the machines are repaired. This concept mirrors situations in transportation, where a retailer may be unable to receive one or more HDVs due to unforeseen events within the warehouse. Consequently, HDVs must adjust their routes to ensure timely deliveries when the goods can be received and stocked.

Finally, the integration between the autonomous vehicle and manufacturing sectors was achieved by grouping AGVs into platoons. The work in Section 4.3 represents the initial implementation of platooning techniques within a production cell. This setup emulates the concept of digital twins, as discussed in [90] and [91], in the Valenciennes production cell, where AGVs operated on fixed tracks. Consequently, there was no need to plan individual AGV trajectories. Nonetheless, dividing AGVs into platoons represents the first step in integrating concepts from both domains, particularly platooning concepts within manufacturing. For instance, in [70], trajectory planning, extensively studied in the literature of autonomous vehicles, is obtained via centralized control, although vehicles are managed individually, without considering groupings based on production requirements.

Lastly, in Section 4.4, a novel manufacturing system was introduced with the goal of achieving full automation within a production environment. To optimize production processes, trajectory planning for AGVs was implemented using a potential field controller, which represents a valuable approach for trajectory generation, different from other decentralized methods in the literature, as exemplified by works like [71] and [72]. With the potential field controller, AGVs independently compute their trajectories, facilitating trajectory generation while considering multiple objects within the system.
Additionally, AGVs' charging constraints were addressed through a reactive decision-making process that takes the system's state into account. This approach optimizes their schedules for recharging at stations to ensure uninterrupted production. Furthermore, AGVs heading to the same destination were grouped into platoons, which enhanced their arrival timing at machinery. This coordination with machinery availability conserved energy without impacting the production order.
While literature extensively covers AGV motion, such as the centralized approach in [70] or the distributed coordination discussed in [73], and battery charging and discharging modeling as seen in [116] and [117], the integration of these aspects within a manufacturing case study, especially when combined with platooning techniques in specific areas of

the shop floor, remains unexplored. The work in this section, therefore, represents a novel contribution to the field of manufacturing processes.

In conclusion, the results obtained in the three research areas of platooning, scheduling, and manufacturing with platooning techniques are not only satisfactory but also highly promising. These findings lay a robust foundation for a multitude of potential benefits across diverse domains, including road safety, traffic congestion reduction, production optimization, and the ability to effectively address unforeseen challenges in manufacturing processes.
This thesis highlights the potential of autonomous and semi-autonomous systems, like platooning, to revolutionize various aspects across diverse domains. For instance, in the field of transportation, platooning can lead to safer and more efficient road networks, as well as significant reductions in fuel consumption and emissions. Additionally, the optimization of manufacturing processes through platooning technologies can enhance productivity, reduce energy consumption, and improve resource allocation while maintaining competitive performance in the production chain.
Additionally, although product scheduling and AGV trajectory planning were not studied in a combined case, solid groundwork has been laid for future research in this direction. This undoubtedly represents a potential avenue for immediate post-thesis work with a high potential for contributions both to automation and optimization in manufacturing. Nevertheless, the results obtained so far can already find practical applications in various research domains.

Bibliography

[1] Istvan Barabas et al. “Current challenges in autonomous driving”. In: *IOP conference series: materials science and engineering*. Vol. 252. 1. IOP Publishing. 2017, p. 012096.

[2] Shantanu Ingle and Madhuri Phute. “Tesla autopilot: semi autonomous driving, an uptick for future autonomy”. In: *International Research Journal of Engineering and Technology* 3.9 (2016), pp. 369–372.

[3] Araz Taeihagh and Hazel Si Min Lim. “Governing autonomous vehicles: emerging responses for safety, liability, privacy, cybersecurity, and industry risks”. In: *Transport reviews* 39.1 (2019), pp. 103–128.

[4] Eray Yağdereli, Cemal Gemci, and A Ziya Aktaş. “A study on cyber-security of autonomous and unmanned vehicles”. In: *The Journal of Defense Modeling and Simulation* 12.4 (2015), pp. 369–381.

[5] Kyounggon Kim et al. “Cybersecurity for autonomous vehicles: Review of attacks and defense”. In: *Computers & Security* 103 (2021), p. 102150.

[6] Theyazn HH Aldhyani and Hasan Alkahtani. “Attacks to automatous vehicles: a deep learning algorithm for cybersecurity”. In: *Sensors* 22.1 (2022), p. 360.

[7] Alfred Daniel et al. “Procuring cooperative intelligence in autonomous vehicles for object detection through data fusion approach”. In: *IET Intelligent Transport Systems* 14.11 (2020), pp. 1410–1417.

[8] Huihui Pan et al. “Deep learning based data fusion for sensor fault diagnosis and tolerance in autonomous vehicles”. In: *Chinese Journal of Mechanical Engineering* 34.1 (2021), pp. 1–11.

[9] David Isele. “Interactive decision making for autonomous vehicles in dense traffic”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 3981–3986.

[10] Hong Wang et al. “Ethical decision making in autonomous vehicles: Challenges and research progress”. In: *IEEE Intelligent Transportation Systems Magazine* 14.1 (2020), pp. 6–17.

[11] Parag Sewalkar and Jochen Seitz. "Vehicle-to-pedestrian communication for vulnerable road users: Survey, design considerations, and challenges". In: *Sensors* 19.2 (2019), p. 358.

[12] José Javier Anaya et al. "Vehicle to pedestrian communications for protection of vulnerable road users". In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE. 2014, pp. 1037–1042.

[13] Ashish Nanda et al. "Internet of autonomous vehicles communications security: overview, issues, and directions". In: *IEEE Wireless Communications* 26.4 (2019), pp. 60–65.

[14] Philip Polack et al. "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" In: *2017 IEEE intelligent vehicles symposium (IV)*. IEEE. 2017, pp. 812–818.

[15] Jason Kong et al. "Kinematic and dynamic vehicle models for autonomous driving control design". In: *2015 IEEE intelligent vehicles symposium (IV)*. IEEE. 2015, pp. 1094–1099.

[16] Egbert Bakker, Hans B Pacejka, and Lars Lidner. "A new tire model with an application in vehicle dynamics studies". In: *SAE transactions* (1989), pp. 101–113.

[17] I.U. Lomonosov. *Introduction to Railway Mechanics*. Oxford University Press, H. Milford, 1933. URL: `https://books.google.it/books?id=4245AAAAMAAJ`.

[18] Francesco Micheli et al. "NMPC trajectory planner for urban autonomous driving". In: *Vehicle System Dynamics* (2022), pp. 1–23.

[19] Hans Pacejka. *Tire and vehicle dynamics*. Elsevier, 2005.

[20] Zhilu Chen and Xinming Huang. "End-to-end learning for lane keeping of self-driving cars". In: *2017 IEEE intelligent vehicles symposium (IV)*. IEEE. 2017, pp. 1856–1860.

[21] Mingjie Liu et al. "Autonomous lane keeping system: Lane detection, tracking and control on embedded system". In: *Journal of Electrical Engineering & Technology* 16 (2021), pp. 569–578.

[22] Stefano Feraco et al. "Combined lane keeping and longitudinal speed control for autonomous driving". In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 59216. American Society of Mechanical Engineers. 2019, V003T01A018.

[23] Abdallah Said et al. "Local trajectory planning for autonomous vehicle with static and dynamic obstacles avoidance". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2021, pp. 410–416.

[24] Chebly Alia et al. “Local trajectory planning and tracking of autonomous vehicles, using clothoid tentacles method”. In: *2015 IEEE intelligent vehicles symposium (IV)*. IEEE. 2015, pp. 674–679.

[25] Júnior AR Silva and Valdir Grassi. “Clothoid-based global path planning for autonomous vehicles in urban scenarios”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 4312–4318.

[26] Ling Zheng et al. “Bézier curve-based trajectory planning for autonomous vehicles with collision avoidance”. In: *IET Intelligent Transport Systems* 14.13 (2020), pp. 1882–1891.

[27] Long Han et al. “Bezier curve based path planning for autonomous vehicle in urban environment”. In: *2010 IEEE intelligent vehicles symposium*. IEEE. 2010, pp. 1036–1042.

[28] Dequan Zeng et al. “A novel robust lane change trajectory planning method for autonomous vehicle”. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 486–493.

[29] Raul Benitez and Zoran Nenadic. “Robust unsupervised detection of action potentials with probabilistic models”. In: *IEEE Transactions on Biomedical Engineering* 55.4 (2008), pp. 1344–1354.

[30] Alireza Khodayari et al. “A historical review on lateral and longitudinal control of autonomous vehicle motions”. In: *2010 International Conference on Mechanical and Electrical Technology*. IEEE. 2010, pp. 421–429.

[31] Elif Toy Azıziaghdam and Orhan Behiç Alankuş. “Longitudinal control of autonomous vehicles consisting power-train with non-linear characteristics”. In: *IEEE Transactions on Intelligent Vehicles* 7.1 (2021), pp. 133–142.

[32] Andrea Raffin, Michele Taragna, and Michele Giorelli. “Adaptive longitudinal control of an autonomous vehicle with an approximate knowledge of its parameters”. In: *2017 11th International Workshop on Robot Motion and Control (RoMoCo)*. IEEE. 2017, pp. 1–6.

[33] Yunxiao Shan et al. “CF-pursuit: A pursuit method with a clothoid fitting and a fuzzy controller for autonomous vehicles”. In: *International Journal of Advanced Robotic Systems* 12.9 (2015), p. 134.

[34] Myung-Wook Park, Sang-Woo Lee, and Woo-Yong Han. “Development of lateral control system for autonomous vehicle based on adaptive pure pursuit algorithm”. In: *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*. IEEE. 2014, pp. 1443–1447.

[35] Gilles Tagne, Reine Talj, and Ali Charara. “Immersion and invariance control for reference trajectory tracking of autonomous vehicles”. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. IEEE. 2013, pp. 2322–2328.

[36] Armin Norouzi et al. “Lateral control of an autonomous vehicle using integrated backstepping and sliding mode controller”. In: *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics* 233.1 (2019), pp. 141–151.

[37] Yulei Wang et al. “Longitudinal and lateral control of autonomous vehicles in multi-vehicle driving environments”. In: *IET Intelligent Transport Systems* 14.8 (2020), pp. 924–935.

[38] Nastaran Tork, Abdollah Amirkhani, and Shahriar B Shokouhi. “An adaptive modified neural lateral-longitudinal control system for path following of autonomous vehicles”. In: *Engineering Science and Technology, an International Journal* 24.1 (2021), pp. 126–137.

[39] Ehab I Al Khatib, Mohammad Abdel Kareem Jaradat, and Mamoun F Abdel-Hafez. “Low-cost reduced navigation system for mobile robot in indoor/outdoor environments”. In: *IEEE Access* 8 (2020), pp. 25014–25026.

[40] Mohammad Yasser Chuttur and Poomedy Rungen. “Design and Implementation of an Autonomous Wheeled Robot Using IoT with Human Recognition Capability”. In: *2022 3rd International Conference on Computation, Automation and Knowledge Management (ICCAKM)*. IEEE. 2022, pp. 1–5.

[41] Richard Pates, Carolina Lidström, and Anders Rantzer. “Control using local distance measurements cannot prevent incoherence in platoons”. In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE. 2017, pp. 3461–3466.

[42] Xin Chang et al. “Analysis on traffic stability and capacity for mixed traffic flow with platoons of intelligent connected vehicles”. In: *Physica A: Statistical Mechanics and Its Applications* 557 (2020), p. 124829.

[43] Assad Alam et al. “Guaranteeing safety for heavy duty vehicle platooning: Safe set computations and experimental evaluations”. In: *Control Engineering Practice* 24 (2014), pp. 33–41.

[44] Mladen Čičić et al. “Platoon-actuated variable area mainstream traffic control for bottleneck decongestion”. In: *European Journal of Control* 68 (2022), p. 100687.

[45] Philip Koopman and Michael Wagner. “Autonomous vehicle safety: An interdisciplinary challenge”. In: *IEEE Intelligent Transportation Systems Magazine* 9.1 (2017), pp. 90–96.

[46] Monimoy Bujarbaruah et al. “Adaptive MPC for autonomous lane keeping”. In: *arXiv preprint arXiv:1806.04335* (2018).

[47] Xiangguo Liu et al. “Trajectory planning for connected and automated vehicles: Cruising, lane changing, and platooning”. In: *arXiv preprint arXiv:2001.08620* (2020).

[48] Harry Chia-Hung Hsu and Alan Liu. “Kinematic design for platoon-lane-change maneuvers”. In: *IEEE Transactions on Intelligent Transportation Systems* 9.1 (2008), pp. 185–190.

[49] Alexander Kanaris, Elias B Kosmatopoulos, and Petros A Loannou. “Strategies and spacing requirements for lane changing and merging in automated highway systems”. In: *IEEE transactions on vehicular technology* 50.6 (2001), pp. 1568–1581.

[50] Louis A Pipes. “An operational analysis of traffic dynamics”. In: *Journal of applied physics* 24.3 (1953), pp. 274–281.

[51] Yuan-Lin Chen and Chong-An Wang. “Vehicle safety distance warning system: A novel algorithm for vehicle safety distance calculating between moving cars”. In: *2007 IEEE 65th Vehicular Technology Conference-VTC2007-Spring*. IEEE. 2007, pp. 2570–2574.

[52] Chiara Bersani et al. “Rapid, robust, distributed evaluation and control of train scheduling on a single line track”. In: *Control Engineering Practice* 35 (2015), pp. 12–21.

[53] Benjamin Nordell. *Trajectory planning for autonomous vehicles and cooperative driving*. 2016.

[54] GR Janssen et al. “Truck platooning: Driving the future of transportation”. In: (2015).

[55] Kuo-Yun Liang, Jonas Mårtensson, and Karl Henrik Johansson. “When is it fuel efficient for a heavy duty vehicle to catch up with a platoon?” In: *IFAC Proceedings Volumes* 46.21 (2013), pp. 738–743.

[56] Assad Alam et al. “Heavy-duty vehicle platooning for sustainable freight transportation: A cooperative method to enhance safety and efficiency”. In: *IEEE Control Systems Magazine* 35.6 (2015), pp. 34–56.

[57] Sebastian Van De Hoef, Karl Henrik Johansson, and Dimos V Dimarogonas. “Fuel-efficient en route formation of truck platoons”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.1 (2017), pp. 102–112.

[58] Sadayuki Tsugawa, Sabina Jeschke, and Steven E Shladover. “A review of truck platooning projects for energy savings”. In: *IEEE Transactions on Intelligent Vehicles* 1.1 (2016), pp. 68–77.

[59] W Levine and Michael Athans. “On the optimal error regulation of a string of moving vehicles”. In: *IEEE Transactions on Automatic Control* 11.3 (1966), pp. 355–361.

[60] SM Melzer and BC Kuo. “Optimal regulation of systems described by a countably infinite number of objects”. In: *Automatica* 7.3 (1971), pp. 359–366.

[61] Mihailo R Jovanovic and Bassam Bamieh. “On the ill-posedness of certain vehicular platoon control problems”. In: *IEEE Transactions on Automatic Control* 50.9 (2005), pp. 1307–1321.

[62] Toshiyuki Sugimachi et al. “Development of autonomous platooning system for heavy-duty trucks”. In: *IFAC Proceedings Volumes* 46.21 (2013), pp. 52–57.

[63] Jakob Axelsson. “An initial analysis of operational emergent properties in a platooning system-of-systems”. In: *2018 Annual IEEE International Systems Conference (SysCon)*. IEEE. 2018, pp. 1–8.

[64] Sebastian Thormann, Alexander Schirrer, and Stefan Jakubek. “Safe and efficient cooperative platooning”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.2 (2020), pp. 1368–1380.

[65] Mladen Čičić et al. “Coordinating vehicle platoons for highway bottleneck decongestion and throughput improvement”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.7 (2021), pp. 8959–8971.

[66] Siyuan Gong and Lili Du. “Cooperative platoon control for a mixed traffic flow including human drive vehicles and connected and autonomous vehicles”. In: *Transportation research part B: methodological* 116 (2018), pp. 25–61.

[67] R Sygulla, A Bierer, and U Götze. “Material flow cost accounting–proposals for improving the evaluation of monetary effects of resource saving process designs”. In: *Proceedings of the 44th CIRP Conference on Manufacturing Systems*. Vol. 1. 2011.

[68] MN Kripak, ES Palkina, and Ya A Seliverstov. “Analytical support for effective functioning of intelligent manufacturing and transport systems”. In: *IOP Conference Series: Materials Science and Engineering*. Vol. 709. 3. IOP Publishing. 2020, p. 033065.

[69] Stefano Feraco et al. “A local trajectory planning and control method for autonomous vehicles based on the RRT algorithm”. In: *2020 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*. IEEE. 2020, pp. 1–6.

[70] Bai Li et al. “Centralized and optimal motion planning for large-scale AGV systems: A generic approach”. In: *Advances in Engineering Software* 106 (2017), pp. 33–46.

[71] Ivica Draganjac et al. "Decentralized control of multi-AGV systems in autonomous warehousing applications". In: *IEEE Transactions on Automation Science and Engineering* 13.4 (2016), pp. 1433–1447.

[72] Guillaume Demesure et al. "Decentralized motion planning and scheduling of AGVs in an FMS". In: *IEEE Transactions on Industrial Informatics* 14.4 (2017), pp. 1744–1752.

[73] Jianbin Xin et al. "Model Predictive Path Planning of AGVs: Mixed Logical Dynamical Formulation and Distributed Coordination". In: *IEEE Transactions on Intelligent Transportation Systems* (2023).

[74] Jiabin Luo, Yue Wu, and André Bergsten Mendes. "Modelling of integrated vehicle scheduling and container storage problems in unloading process at an automated container terminal". In: *Computers & Industrial Engineering* 94 (2016), pp. 32–44.

[75] Fred Hanssmann and Sidney W Hess. "A linear programming approach to production and employment scheduling". In: *Management science* 1 (1960), pp. 46–51.

[76] Christian D Hubbs et al. "A deep reinforcement learning approach for chemical production scheduling". In: *Computers & Chemical Engineering* 141 (2020), p. 106982.

[77] Xiaohan Wang et al. "Logistics-involved task scheduling in cloud manufacturing with offline deep reinforcement learning". In: *Journal of Industrial Information Integration* 34 (2023), p. 100471.

[78] Antonio Reyes Moro, Hongnian Yu, and Gerry Kelleher. "Hybrid heuristic search for the scheduling of flexible manufacturing systems using Petri nets". In: *IEEE Transactions on Robotics and Automation* 18.2 (2002), pp. 240–245.

[79] Ihsan Sabuncuoglu and Omer Batuhan Kizilisik. "Reactive scheduling in a dynamic and stochastic FMS environment". In: *International journal of production research* 41.17 (2003), pp. 4211–4231.

[80] Yoann Fouquet et al. "Flow Adjustment-a Flexible Routing Strategy for Demand Protection Against Multiple Partial Link Failures". In: *INFOCOMP: International Conference on Advanced Communications and Computation.* Vol. 4. 2014, pp. 83–90.

[81] Dritan Nace et al. "A polynomial multicommodity flow problem with difficult path generation". In: *2011 3rd International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT).* IEEE. 2011, pp. 1–5.

[82] Yaping Fu et al. "Multiobjective modeling and optimization for scheduling a stochastic hybrid flow shop with maximizing processing quality and minimizing total tardiness". In: *IEEE Systems Journal* 15.3 (2020), pp. 4696–4707.

[83] George Chryssolouris and Velusamy Subramaniam. “Dynamic scheduling of manufacturing job shops using genetic algorithms”. In: *Journal of Intelligent Manufacturing* 12 (2001), pp. 281–293.

[84] N Jawahar, P Aravindan, and SG Ponnambalam. “A genetic algorithm for scheduling flexible manufacturing systems”. In: *The International Journal of Advanced Manufacturing Technology* 14 (1998), pp. 588–607.

[85] Libing Wang et al. “Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning”. In: *Computer Networks* 190 (2021), p. 107969.

[86] Yu Du et al. “Knowledge-based reinforcement learning and estimation of distribution algorithm for flexible job shop scheduling problem”. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* (2022).

[87] Soroush Fatemi-Anaraki et al. “Scheduling of multi-robot job shop systems in dynamic environments: mixed-integer linear programming and constraint programming approaches”. In: *Omega* 115 (2023), p. 102770.

[88] Ziyan Zhao et al. “Dual-objective mixed integer linear program and memetic algorithm for an industrial group scheduling problem”. In: *IEEE/CAA Journal of Automatica Sinica* 8.6 (2020), pp. 1199–1209.

[89] Jose-Fernando Jiménez. “Dynamic and hybrid architecture for the optimal reconfiguration of control systems: application to manufacturing control”. PhD thesis. Université de Valenciennes et du Hainaut-Cambresis, 2017.

[90] Yilin Fang et al. “Digital-twin-based job shop scheduling toward smart manufacturing”. In: *IEEE transactions on industrial informatics* 15.12 (2019), pp. 6425–6435.

[91] Khalil Tliba et al. “Digital twin-driven dynamic scheduling of a hybrid flow shop”. In: *Journal of Intelligent Manufacturing* 34.5 (2023), pp. 2281–2306.

[92] Jose-Fernando Jimenez et al. “Analysing the impact of rescheduling time in hybrid manufacturing control”. In: *Service Orientation in Holonic and Multi-Agent Manufacturing: Proceedings of SOHOMA 2016*. Springer. 2017, pp. 225–236.

[93] Alberto Villalonga et al. “A decision-making framework for dynamic scheduling of cyber-physical production systems based on digital twins”. In: *Annual Reviews in Control* 51 (2021), pp. 357–373.

[94] Abtin Nourmohammadzadeh and Sven Hartmann. “The fuel-efficient platooning of heavy duty vehicles by mathematical programming and genetic algorithm”. In: *Theory and Practice of Natural Computing: 5th International Conference, TPNC 2016, Sendai, Japan, December 12-13, 2016, Proceedings 5*. Springer. 2016, pp. 46–57.

[95] Dahlia Sam, Cyrilraj Velanganni, and T Esther Evangelin. "A vehicle control system using a time synchronized Hybrid VANET to reduce road accidents caused by human error". In: *Vehicular communications* 6 (2016), pp. 17–28.

[96] Shaofeng Lu, Stuart Hillmansen, and Clive Roberts. "A power-management strategy for multiple-unit railroad vehicles". In: *IEEE transactions on vehicular technology* 60.2 (2010), pp. 406–420.

[97] Cecilia Pasquale et al. "A new Micro-Macro METANET model for platoon control in freeway traffic networks". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 1481–1486.

[98] Apostolos Kotsialos et al. "Traffic flow modeling of large-scale motorway networks using the macroscopic modeling tool METANET". In: *IEEE Transactions on intelligent transportation systems* 3.4 (2002), pp. 282–292.

[99] Ather Gattami, Anders Rantzer, and Bo Bernhardsson. "Robust team decision theory". In: *IEEE Transactions on Automatic Control* 57.3 (2012), pp. 794–798.

[100] Vishakha Vijay Patel. "Ziegler-Nichols Tuning Method: Understanding the PID Controller". In: *Resonance* 25.10 (2020), pp. 1385–1397.

[101] A. Gattami, B. M. Bernhardsson, and A. Rantzer. "Robust Team Decision Theory". In: *IEEE Transactions on Automatic Control* 57.3 (2012), pp. 794–798. DOI: `10.1109/TAC.2011.2168071`.

[102] A. Gattami and B. Bernhardsson. "Minimax Team Decision Problems". In: *2007 American Control Conference*. 2007, pp. 766–771. DOI: `10.1109/ACC.2007.4282858`.

[103] Alessandro Bozzi et al. "Real-time Robust Trajectory Control for Vehicle Platoons: A Linear Matrix Inequality-based Approach." In: *ICINCO*. 2021, pp. 410–415.

[104] Alessandro Bozzi, Roberto Sacile, and Enrico Zero. "Proportional Integral Derivative Decentralized Control vs Linear Quadratic Tracking Regulator in Vehicle Overtaking within a Platoon". In: (2022).

[105] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. "On a formal model of safe and scalable self-driving cars". In: *arXiv preprint arXiv:1708.06374* (2017).

[106] Bernd Gassmann et al. "Towards standardization of av safety: C++ library for responsibility sensitive safety". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 2265–2271.

[107] Stephen Boyd. *EE363 Review session 1: LQR, Controllability and Observability*. `https://stanford.edu/class/ee363/sessions/s1notes.pdf`. [Online; accessed 19-July-2008]. 2008.

[108] A Bozzi et al. "A hierarchical control scheme to improve the travel performance of truck platoons in freeways". In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2022, pp. 2063–2068.

[109] George Ellis. “Chapter 6 - Four Types of Controllers”. In: *Control System Design Guide (Fourth Edition)*. Ed. by George Ellis. Fourth Edition. Boston: Butterworth-Heinemann, 2012, pp. 97–119. ISBN: 978-0-12-385920-4. DOI: https://doi.org/10.1016/B978-0-12-385920-4.00006-0. URL: https://www.sciencedirect.com/science/article/pii/B9780123859204000060.

[110] Andreas S Schulz. “Scheduling to minimize total weighted completion time: Performance guarantees of LP-based heuristics and lower bounds”. In: *Integer Programming and Combinatorial Optimization: 5th International IPCO Conference Vancouver, British Columbia, Canada, June 3–5, 1996 Proceedings 5*. Springer. 1996, pp. 301–315.

[111] Jean-Luc Maire, Vincent Bronet, and Maurice Pillet. “Benchmarking: methods and tools for SME”. In: *Benchmarking: An International Journal* 15.6 (2008), pp. 765–781.

[112] Alessandro Bozzi et al. “Reliability Evaluation of Emergent Behaviour in a Flexible Manufacturing Problem”. In: *2023 18th Annual System of Systems Engineering Conference (SoSe)*. IEEE. 2023, pp. 1–7.

[113] Roger Stafford. *Random Vectors with Fixed Sum*. https://www.mathworks.com/matlabcentral/fileexchange/9700-random-vectors-with-fixed-sum. 2023.

[114] Aaron French et al. *Multivariate analysis of variance (MANOVA)*. 2008.

[115] *Safelog AGVX1 datasheet*. https://www.safelog.de/en/wp-content/uploads/sites/2/2022/08/Datasheet_AGV_X1_english_preliminary.pdf.

[116] Moussa Abderrahim et al. “Manufacturing 4.0 operations scheduling with AGV battery management constraints”. In: *Energies* 13.18 (2020), p. 4948.

[117] Xiangnan Zhan et al. “Study on AGVs battery charging strategy for improving utilization”. In: *Procedia CIRP* 81 (2019), pp. 558–563.