Università degli Studi di Modena e Reggio Emilia

DOTTORATO DI RICERCA IN
INGEGNERIA DELL'INNOVAZIONE INDUSTRIALE

Ciclo XXVIII

Traffic Coordination for AGV Systems: an Ensemble Modeling Approach

Candidato: **Valerio Digani**

Direttore della Scuola:
Prof. Mauro Dell'Amico

Relatore:
Prof. Cristian Secchi

Esame finale anno 2016

Reggio Emilia, Italy, March 2016.

To Ester
To my Family
You supported me, encouraged and
helped to be the person I am now.

Abstract

This thesis deals with an ensemble strategy for optimizing the overall performance of multi-robot systems. Specifically, methodologies are presented for the coordination, optimization and traffic management of a fleet of automated guided vehicles (AGV) operating in industrial environments.

This dissertation presents first an algorithm for the automatic creation of a roadmap for AGV systems which maximizes the connectivity, the redundancy and the coverage. The roadmap is built in such a way that the environment is filled by as roads as possible whose directions are assigned by maximizing the connectivity of the associated graph.

A traffic coordinator is then developed in such a way that the optimal performance is guaranteed when it is used with a roadmap generated with the previous method. The coordination is performed by means of both decentralized and centralized control policies. The former is based on a priority scheme for the resource allocation, the latter is performed by modeling the coordination problem as a quadratic programming problem (QP) by aiming at minimizing the total time required for the fleet in order to accomplish its tasks.

Along with the traffic coordinator, a hierarchical 2-layers control architecture is developed. The architecture exploits two layers to manage the problem. A layer is a topological graph representing the roadmap where each node is an area of the roadmap called sector. Then the other layer represents the actual roadmap within each sector. In this way, the overall scenario is modeled as a lumped parameter model where the traffic congestions and then the complexity are bounded in specific areas.

Furthermore a probabilistic dynamic model of the traffic is introduced. The model is used to predict the evolution of the traffic in a future horizon. This information is then exploited by a traffic-based planner which coordinated the vehicles in order to minimize the total

arrival time.

The methodologies are supported by simulations and experiments in real world scenario, specifically in real automatic warehouses.

Sommario

Il tema della tesi consiste nello sviluppo di strategie globali per l'ottimizzazione delle performance in sistemi multi-robot. In particolare, si sono analizzati e sviluppati metodi per la coordinazione, l'ottimizzazione e la gestione del traffico di una flotta di automated guided vehicles (AGV) operanti in ambienti industriali.

Il lavoro di tesi ha visto per primo lo sviluppo di un algoritmo per la creazione automatica delle roadmap per sistemi AGV, massimizzandone la connettività, la ridondanza e l'indice di copertura. La roadmap generata è tale che l'ambiente risulti riempito dal massimo numero possibile di strade, mentre le direzioni sono assegnate ottimizzando la connettività del grado associato.

In seguito, è stato sviluppato un coordinatore del traffico che garantisce ottime prestazioni qualora usato per coordinare la flotta su roadmap generate con il metodo automatico. Si sono analizzati due metodi di coordinazione tra veicoli: decentralizzato e centralizzato. Il primo si basa su uno schema di priorità con lo scopo di allocare risorse, nel secondo il problema è affrontato tramite una modellazione di programmazione quadratica (QP) dove si minimizza il tempo totale per completare le missioni della flotta.

A fianco del coordinatore, si è sviluppata una architettura di controllo basata su due livelli. La natura gerarchica permette così di ridurre la complessità totale del problema. Il primo livello (più astratto) è un grafo topologico rappresentate la roadmap e l'ambiente. Ogni nodo è una area specifica della mappa chiamata *settore*. Il secondo livello rappresenta la roadmap vera e propria all'interno dei singoli settori. Tale metodo permette di modellare il sistema a parametri concertati e il traffico è così racchiuso solo in alcune aree.

In fine, si è sviluppato un modello probabilistico del traffico tra i vari settori, che permette di predirne l'evoluzione all'interno di un certo

orizzonte temporale. Ciò è inglobato in un pianificatore di percorsi che tiene in conto lo stato del traffico e coordina la flotta minimizzandone il tempo totale necessario per il completamento delle missioni.

A supporto della tesi, è stata condotta una intensa campagna di validazione sia tramite simulazioni che esperimenti relativi a scenari realistici, nello specifico, a magazzini automatici.

Acknowledgements

This thesis is the result of several years of work, and I need to thank who made it possible.

I would like to thank my advisor, Prof. Cristian Secchi, for giving me this opportunity, and his help in these years and for having always supported me.

Many thanks go to Prof. Cesare Fantuzzi for his advices and his help during this period and when I had to do difficult choices.

A huge thanks goes to Dr. Lorenzo Sabattini, for having constantly advice and help me.

I would like to thank also Prof. M. Ani Hsieh for giving me the opportunity of having an amazing experience in Philly at Drexel University.

Thanks to all the ArsControl staff, thank you guys for having share many moments with me.

I really need to thank my Family and Ester, Thanks for being there.

Contents

Chapter 1

Introduction

In the recent years the necessity of increasing the goods production flow in manufacturing plants while decreasing the costs is becoming more and more crucial. In the last decades, manufacturing plants have been largely automated in order to mainly reduce costs and avoid unsafe work condition and robotics has contributed and is contributing to this purpose. Factory logistics is crucial for the overall production flow and its weaknesses affect the production efficiency and the quality of goods delivery, in particular in terms of product traceability. Futhermore bottlenecks and problems in warehouse logistics impact heavily on factory competitiveness in the market. For this purpose, Automated Guided Vehicles (AGVs) have been introduced since the 1950s for automatic material handling systems because of their flexibility and efficiency. This Thesis deals with the development of a global strategy for optimizing an AGV system. Very often, in the common industrial practice, the roadmap (namely the paths on which the AGVs move) and the coordination strategies are separately developed, without considering their existing relationship. The way the AGVs move is intrinsically dependent on how the roadmap is built, and vice-versa. Moreover, manual rules are also commonly used for avoiding local issues, i.e. deadlocks at intersections, etc.

This chapter gives an overview of the AGVs system and motivations and objectives of this dissertation are treated in more detail. Sec. 1.1 describes briefly current AGV systems, Sec. 1.3 reports references and notions about multi-robot system and the objective and motivations of the presented work are reported in Sec 1.4.

1.1 Automated Guided Vehicles

An automated guided vehicle (AGV) is a driverless transport system used for movement of materials. Generally speaking, it is a mobile robot which can transport

goods from a location to another in a warehouse. Clearly, the specifications of AGVs differ per environment. To transport a container, the capacity of an AGV should at least be equal to 40 tonnes. Less capacity is required for the transport of pallets at warehouses. Typically the main components in an AGV are:

- localization system: typically laser-based systems or magnetic systems
- navigation system: a navigation laser as well as encoders provide the position of the vehicle which can then navigate in the environment
- safety system: proximity lasers or bumpers to avoid collisions
- communication system: all the AGVs have to communicate with a control unit

A typical AGV forks is shown in fig. 1.1.

Figure 1.1: Automated Guided Vehicles (AGVs)

1.2 AGV System

Modern logistics facilities such as warehouses, distribution centers and production plants are the typical scenarios for implementing AGV systems. A portion of a warehouse where AGVs are exploited is shown in Fig. 1.2. AGV systems have been extensively studied in the literature: a comprehensive survey is presented in [1], where authors describe the main technologies adopted for localization and guidance of AGVs in industrial environments. Thus, AGVs are generally exploited for goods transportation, that is for moving batches of goods (pallets or boxes) from one location to another one, according to the requests from the logistics system [2].

In the following paragraphs the AGV system scenario is described.

Figure 1.2: AGV system exploited in industrial logistics

1.2.1 Scenario

In automatic warehouses, AGVs usually pick up pallets of goods from an automated production line. The pallets have to be brought to the shipment area, usually composed by empty trucks. Sometimes the pallets cannot be shipped directly, but need to be stored in a warehouse. The storage areas are replenished with incoming loads, which are outputted by the production lines or delivered by trucks during the day.

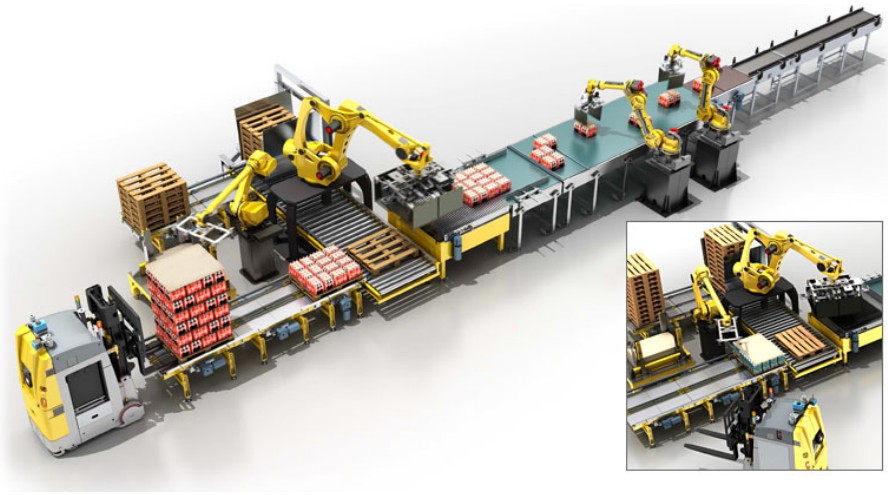


Figure 1.3: Example of end-line in automatic warehouses

The warehouse management system (WMS) then allocates the orders to locations within the storage areas, and order picking routes are determined. AGVs have to travel through the aisles between the racks in the storage area to visit the locations to collect the orders. These routes are determined with objective criteria such as:

- minimize the travel time

- minimize the waiting time caused by traffic congestion.

The sequence of the executed orders depends on their priorities, whether the orders involves single or multiple pallet, etc. It is worth noting that the the pick times and travel times of AGVs are stochastic due to the following reasons:

- acceleration/deceleration effects
- failure of equipment
- unexpected obstacles along the paths
- traffic congestion due to unforeseen events

The drop off instants of the loads are also stochastic. A real AGV system in industrial environment is then a non-deterministic scenario. The main characteristics of an AGV system will now be analyzed in details.

Plant Installation The installation phase for an AGV system is a complex set of operations, that need to be carried out for the system to be able to work in an autonomous manner. Roughly speaking, given the geometric layout of the plant, and given the expected flows of material during the plant operation, the infrastructure for the localization system has to be set up, and the roadmap has to be defined. Specifically, we consider laser guided vehicles. Namely, each AGV is able to localize itself measuring its relative distance from some previously mapped artificial landmarks. These landmarks are made of reflective material, and distances are measured by means of a laser scanner, placed on the top of each AGV. A precise knowledge of the map of landmarks is mandatory for obtaining a highly precise localization. Moreover, the position of the landmarks themselves has a great influence on the localization accuracy.

The *roadmap* is the set of paths along which the AGVs can travel and it can be divided in small portions called *segments*. The roadmap has to be defined such that a path that connects each pair of operation positions exists. Moreover, each path is defined in order to avoid collisions with the infrastructure, and among the AGVs. For each segment of the roadmap, the maximum allowed speed is defined, in such a way that the safety devices (typically laser scanners) are able to effectively detect unforeseen obstacles, and appropriately stop the AGV. An example of roadmap defined in a portion of a plant is represented in Fig 1.4.

The design of the roadmap is generally performed manually with a CAD software and requires a highly specialized operator. In fact the roadmap design impacts on AGV traffic and then defines the system efficiency.

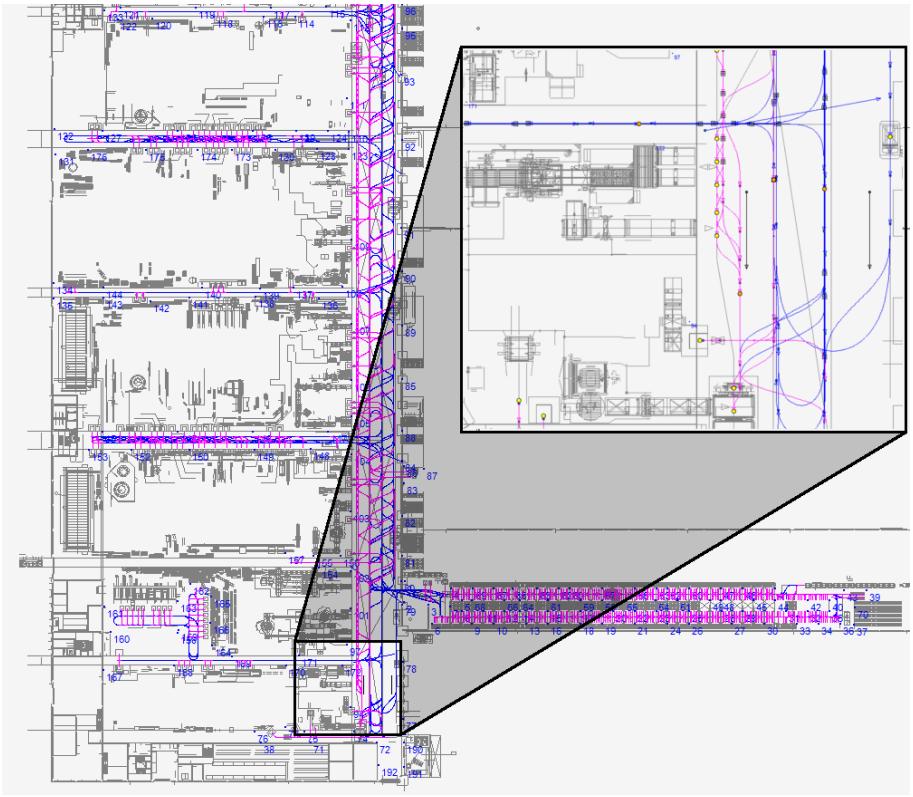


Figure 1.4: Complete and detailed portion of an AGV roadmap

Mission Assignment Each AGV has to accomplish a task that consist in moving goods among different locations of the environment, for instance from the production area to the storage area. The process of selecting and assigning transport tasks to vehicles is called dispatching. Each transportation task is defined as a mission. The AGVs are in general dispatched on-line, i.e. based on real-time information, since the uncertainty of the load release and delivery times makes vehicle dispatching beforehand (scheduling) very hard. The locations where loading and unloading operations are performed are generally referred to as operation points. The position and the characteristics of each operation point need then to be mapped with high precision, in order to ensure that loading and unloading operations can be performed in an effective manner.

In order to optimize the overall performance of the system, it is necessary perform the assignment in an optimized manner, that means minimizing the overall completion time. Mainly utilized assignment methods consider the mission assignment problem in a static manner: roughly speaking, each mission is assigned to the closest AGV. Common algorithms are used for such a scenario, for instance the Hungarian algorithm, presented in [3, 4], provides an optimal solution to the assignment

problem in polynomial time.

AGV coordination The motion of the AGVs is coordinated inside the warehouse in order to ensure completion of all missions in an efficient manner. If the traffic is not properly treated, congestion, deadlocks or even collisions can take place. These situations can block part of the system and they can require to stop the AGVs to allow the intervention of qualified personnel for a manual restart drastically reducing the performance of the system. The traffic control problem is further complicated by the presence of unpredictable events that can take place in automatic warehouses. For example, a pallet may fall during the transportation or an AGV can suddenly stop because of a fault. These events produce some unplanned static obstacles along the routes tracked by the vehicles. Furthermore, the AGVs are working in a dynamic environment populated by other moving obstacles (i.e. manual forklifts) whose trajectories cannot be predicted while the safety has to be always guaranteed. The path planning and traffic coordination is a very complex problem, whose complexity grows exponentially with the number of vehicles. The complexity is reduced by using the roadmap where the AGVs can move. It is worth noting that, in general, the design of the roadmap and the design of the coordination algorithm both contribute to the overall efficiency of the system. Typically, specific features of the roadmap are handled by adding exceptions to the coordination algorithm, in the form of traffic rules [5]. Moreover, while roadmaps are a very effective manner of reducing the computational resources needed for traffic management, constraining the motion of the AGVs on a finite set of roads severely reduces the flexibility of the system. For instance, if an obstacle appears on an AGV's road, it is necessary to re-plan the path to circumvent the obstacle. When an alternative path is not available, traffic jams might appear, that can be solved only with the intervention of an operator, to manually remove the obstacle.

Nowadays these autonomous systems have a market share of about few thousands vehicles sold every year and they are not yet ready to be widespread in manufacturing plants. In fact, some open problems still remain, such as safety, efficiency and plant installation costs. Therefore, innovations to address weaknesses of AGVs and automated warehouse systems will boost capabilities of these logistic solutions bringing them toward a pervasive diffusion in modern factories.

1.3 Multi-Robot Systems

In the last years, multi-robot systems have received more and more attention by the robotics community. AGV systems represent a typical industrial application of multi-robot scenarios. Multi-robot systems can be classified in several ways, in this section a collection of the approaches presented in literature is given following the classifications given by [6] and [7].

1.3.1 Centralized versus Decentralized

This characterization is based on the three aspects: the architecture of the system, the temporal scope of the planning and the decomposition of the coordination problem. The architecture can be centralized or decentralized (or distributed). In the first case, a central unit gathers the data of all the robots and then it decides the motion of each vehicle. The main advantage of these approaches is that they allow to obtain a very efficient coordination and, in principle, even the optimal one (with respect to a chosen functional). For this reason, fleets of AGVs in industrial applications are generally coordinated by a centralized supervisor (the control center) which manages all the information coming form the Warehouse Management System (WMS) and from the environment. The control center handles the coordination of the fleet, solving a multi-robot path planning problem (see e.g [8, 9]). The main drawback of these algorithms is the computational burden since, in general, the dimension of the multi-robot space may be very high and then it may require a considerable computational resources. Furthermore, centralized approaches represent single point of failure systems and then the robustness of the system can be limited.

Traffic problems have been also treated using decentralized architectures. In decentralized architectures, each vehicle decides its motion based on local information. In these methods, each robot autonomously determines its routes, dissolving the conflicts and collecting information from other robots [10, 11]. Decentralized techniques are generally faster than centralized ones, but they present several drawbacks. While existing work such as [12, 13], provides deadlock-free strategies, decentralized approaches can fail to find valid paths for all robots and feasible solutions may be far from optimal. As such, decentralized strategies may not be suitable for many industrial applications where efficiency is paramount.

1.3.2 Coupled versus Decoupled

The coordination strategies can be classified in: coupled approaches and decoupled approaches. Coupled approaches search the solution of the motion planning problem in a composite coordination space, which is formed by the Cartesian product of the configuration spaces of the individual vehicles. Namely these approaches aggregate all the individual robots into one large composite system and apply single-robot motion planning algorithms for solving the coordination problem. Much of classical motion planning techniques for exact motion planning, randomized motion planning and their variants would apply directly [14, 15, 16, 7]. Except from their very high complexity, this kind of approaches are non-robust to any contingencies arising during the system operation. They require to discard the plans every time the coordination problem must be updated (e.g., a new mission is assigned to a vehicle). These approaches are generally characterized by a significant computational burden so that their application is often limited to simple problem settings involving two or three vehicles. For example, if we have m robots with k degrees of freedom each, a coupled planner will have a time complexity exponential in $m \times k$. This means that, assuming we do not consider the dimension of the composite configuration space as a constant, the multi-robot path planning problem is *PSPACE-complete* (see, e.g., [17]).

Decoupled approaches face the complexity of the coordination, by breaking the problem into two distinct phases: path planning and motion coordination. During the first phase a path for each vehicle is planned without considering the presence of other vehicles. In the second phase the velocity profile of each vehicle along its path is computed. These approaches are more suitable than centralized ones for dealing with coordination problems involving a big number of vehicles. Decoupled approaches are typically divided into two broad categories: *prioritized planning* and *path coordination*. *Prioritize planning* considers the motions of the robots one at a time, in a priority order. In particular, the path of the robot i is computed by considering the $i-1$ paths of the other robots as moving obstacles. *Path coordination*, on the other hand, first plans independent paths for the robots separately, then seeks to plan their velocities so as to avoid collisions along those paths.

1.3.3 Roadmap-based Approach

In general the dimension of the multi-robot space may be very high or even intractable. One way to reduce the search space is to weakly constrain the allowable paths that robots can follow by limiting the motion of the robots to lie on *roadmaps*

in the environment. Intuitively, roadmaps are akin to automotive highways, where robots move from their starting position to a roadmap, move along the roadmap to the proximity of the goal, and then move off the roadmap to the specific goal location.

Several strategies can be found in the literature for the definition of a roadmap. Generally speaking most roadmap based algorithms have been designed for motion planning for a single robot in a static environment and are generated based on random sampling techniques [15]. Several algorithms, known collectively as probabilistic roadmap methods (PRMs), have been shown to perform well in a number of practical situations [14, 18]. These methods run quickly and are easy to implement, but they can perform poorly in some situations [19]. Furthermore there are several extensions based on the PRM algorithm. Visibility-PRMs [20, 15] try to build small roadmaps avoiding redundant elements, instead Reachability Roadmap Methods (RRM) emphasize the coverage and the maximal connectivity of the free space $\mathcal{C}_{free}$ [21] by creating small, resolution complete roadmaps. The approach described in [22] aims at providing high-clearance paths by retracting the roadmap to the medial axis [23]. Some algorithms have been proposed to extend the roadmap based algorithms to dynamic environments [24] and multiple agents [25]. However none of these algorithms keeps into account how the robots will be actually coordinated.

As illustrated in [26], in the vast majority of modern automatic warehouses, AGVs are constrained to move on a roadmap (see for instance fig. 1.2). Up to sixty AGVs can travel in an automatic warehouse and the way the roadmap is designed tremendously affects the way traffic can be managed and, consequently, the efficiency of the overall system. In the current industrial practice, given the layout of a plant, the roadmap is manually designed. This process is very time consuming and the achieved set of paths can be quite far from the best one.

1.4 Contribution and thesis outline

This Thesis focuses on the coordination of AGVs in an industrial environment. The management of a fleet of vehicles is faced by considering all the aspects concerning the overall system. In other words, the problem is addressed in terms of performance of the system, coordination strategy, control architecture and optimization method. The motivation of the dissertation can be found in the more and more increasing demands of automatized industrial warehouses. The typical example in this context is the Kiva System solution (now Amazon Robotics), where hundreds of robot move in the same warehouse [27]. Generally speaking, most of solutions presented in the

robotics community can focus on specific aspects of the problem (e.g. path planning, coordination, communication, etc.) without considering the interconnections among them. For this reason, this thesis faces the problem in a different manner: all the aspects of an AGV system are put together for considering an ensemble model of the problem. A solution of the traffic coordination problem is then provided.

Each chapter faces a specific aspect of the ensemble traffic coordination approach. Each of them starts with an introduction section, that includes a detailed analysis of the state of the art, based on the literature review. The organization of the thesis is described in the follow.

Chapter 2 describes an algorithm for the *automatic generation of a roadmap* for AGV systems. Such a roadmap is build in order to be maximal connected, redundant and to cover all the free space. Simulation on real plant form automated warehouses are also dealt.

Chapter 3 proposes a hierarchical architecture to face the coordination problem. In particular, a 2-layers approach is proposed in order to reduce the complexity of large-scale multi-robot coordination problem. A basic coordination algorithm is also reported.

An optimized coordination algorithm is shown in **Chapter 4**. The coordination is faced by modeling the problem as a quadratic optimization problem (QP) whose optimization variables are the AGVs' velocities. Experiments and simulations are also discussed.

Chapter 6 deals with traffic models for AGV systems. In particular, first a simple static model is proposed and the task assignment problem is formulated on this. Subsequently a probabilistic dynamic traffic model is detailed described. The latter model is able to predict the evolution of the traffic which can be used to plan better paths for the AGVs. Finally, in the last chapter some conclusions are fulfilled.

Chapter 2

Roadmap Generation Algorithm

In this chapter an algorithm for the automatic creation of a roadmap for AGV systems is described. The algorithm computes a roadmap in such a way that the coverage, the connectivity and the redundancy of the paths are maximized. In this way the flexibility and the efficiency of the AGV system can be increased. The proposed approach is validated by means of a comparison with standard industrial approach for roadmap generation in real plants.

2.1 Introduction

As illustrated in [26, 28], and summarized in the Chapter 1, in the vast majority of modern automatic warehouses, AGVs are constrained to move along a set of (virtual) paths and this set is usually called roadmap. Hundreds of robots can travel in an automatic warehouse and the way the roadmap is designed tremendously affects the way traffic can be managed and, consequently, the efficiency of the overall system. For instance, *Kiva Systems* proposes a solution where hundred of robots can move in the warehouse in grid-based approach [27]. However, in the current industrial practice for AGV solutions, given the layout of a plant, the roadmap is manually designed. Expert personnel is then required to design the roadmap by means of CAD software. This process is very time consuming and the achieved set of paths can be quite far from the best one. Thus, an automatic procedure for designing a roadmap which dramatically decreases the installation time and the cost of an AGV system is proposed.

A roadmap can be represented by a topological graph spanning the free space [29]. Most roadmap based algorithms have been designed for motion planning for a single robot in a static environment and are generated based on random sampling techniques [15].

Lots of works are concerned with methods to build a roadmap. Several algorithms, known collectively as probabilistic roadmap methods (PRMs), have been shown to perform well in a number of practical situations [14, 18]. The idea behind these methods is to create a graph of randomly generated collision-free configurations [15]. Connections between these configurations are made by a simple and fast local planning method. Global planning is then carried out on the roadmap. These methods run quickly and are easy to implement, but they can perform poorly in some situation [19]. e.g., when paths are required to pass through narrow passages in configuration space [30]. Furthermore there are several extensions based on the PRM algorithm. Visibility-PRMs [20, 15] try to build small roadmap avoiding redundant elements by connecting only the new nodes to useful ones as shown in [31]. Thus Visibility-PRMs capture the free space connectivity with a roadmap much smaller than Basic-PRM while maintaining the same coverage of the free space [32]. Although this approach prunes the roadmap a lot, it is slower than other variants of PRM. Reachability Roadmap Methods (RRM) have merged the advantages of Basic-PRMs and Visibility-PRMs [33, 34]. RRMs emphasize the coverage and the maximal connectivity of the free space $\mathcal{C}_{free}$ [21]. This technique creates small, resolution complete roadmaps. Nonetheless many techniques generate low quality paths. The approach described in [22] aims at providing high-clearance paths by retracting the roadmap to the medial axis [23]. Some algorithms have been proposed to extend the roadmap based algorithms to dynamic environments [24] and multiple agents [25]. However none of these algorithms keeps into account how the robots will be actually coordinated.

Approaches [35, 36] show a combined planning methods based on network flow and medial axis. In particular the former provides an interesting parallelism among network flow problem on a graph and multi-agent path planning problem, but it fails in providing an applicable solution. The other work focuses on providing an integer linear programming (ILP) planning method which minimizes the average of the arrival times of all the agents over the maximum number of agents contained in the same edge at a given time stamp (capacitated graph). Furthermore the graph is built on the medial axis of the free space. This approach is designed for large groups and considers some restricted constraints and assumptions which are not allowed in the AGV system scenario.

In the following sections the details of the proposed algorithm will be discussed. The algorithm, given the layout of the plant, provides a roadmap such that the coverage, the connectivity and the redundancy are maximized. The generated roadmap

is suitable to be used with the traffic coordination strategy reported in [37] where the coordination and the roadmap generation problems are treated as a whole. An evidence of this claim is that the performance of the coordination algorithm is strongly related to the structure of the roadmap where the coordination takes place.

2.1.1 Outline

The outline of the Chapter is as follow. Section 2.2 describes the problem and provides some definitions. Section 2.3 shows in detail the algorithm for the generation of the roadmap. Finally Section 2.4 explains the experiments and simulations.

2.1.2 Preliminaries on Medial Axis Transform(MAT)

Since the Medial Axis Transformation will be largely used in the rest of the chapter, a brief introduction is now given.

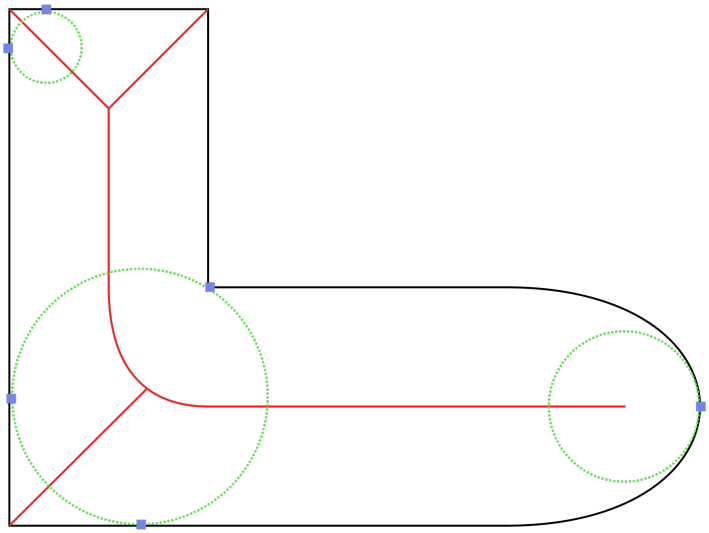


Figure 2.1: Definition of Medial Axis Transformation: red line is the medial axis

The medial axis of an object is the set of all points having more than one closest point to the object's boundary. In the 2D case, the following facts hold [23].

- Given a planar region B bounded by a curve L, the medial axis of B is the set of the centers of the circles that are contained in B and that are tangent to L in at least two points.
- The medial axis of a simple polygon is a tree whose leaves are the vertices of the polygon, and whose edges are either straight segments or arcs of parabolas.

- The medial axis together with the associated radius function of the maximally inscribed discs is called the *medial axis transform* (MAT)

Let $\mathcal{Q}$ be the set of segments of line that compose the medial axis. MAT is a complete shape descriptor and it can be used for reconstructing the shape of the original domain.

An algorithm for computing the medial axis of polygonal regions is proposed in [38]. It runs in *O(N log N)* time, where N is the number of segments of the polygon. Another approach for computing the medial axis for non polygonal regions can be found in [39]. The planar region is discretized into a set of μ cells and the MAT is found using a two-step dynamic program that runs in $O(\mu)$.

2.2 Problem Statement

The approach aims at filling an industrial warehouse with feasible paths. Given the layout of the plant, the configuration space $\mathcal{C}$ is formally defined as the set of all the possible positions of every point in the system. It is discretized by building a grid map and $\mathcal{C}_{free}$ indicates the portion of $\mathcal{C}$ that is not occupied by obstacles. Therefore, hereafter the free space $\mathcal{C}_{free}$ to be constituted of a finite number of cells, that represent the discretization of all the configurations in the free space. The cells of $\mathcal{C}_{free}$ define then a grid map. A euclidean Distance Transformation (DT) [40] is then applied on the grid map in order to discretize the distance from the obstacles. In this way a value indicating the euclidean distance form the obstacles is applied at each cell of the grid map.

A *roadmap* is a set of routes, composed of distinguished elements called segments (see Fig. 2.2). The AGVs are constrained to follow the roadmap and its segments. More formally, a roadmap is defined as follows:

Let me introduce the following definition of *feasible path*.

Definition 2.1. Feasible Path *A path $P \in \mathcal{C}$ between two configurations $i, j \in \mathcal{C}$ is feasible, when $P \in \mathcal{C}_{free}$.*

A roadmap can be represented as a graph $\mathcal{G} = (V, E)$, where the vertices V represent the cells $\mathcal{C}_{free}$, and the edges E represent the roads that connect the cells. The following definitions can be introduced:

Definition 2.2. Coverage [22] *Given any two configurations c_1,$c_2 \in \mathcal{C}_{free}$, $\mathcal{G}$ covers $\mathcal{C}_{free}$ if a feasible path between c_1 and c_2 exists in $\mathcal{G}$.*

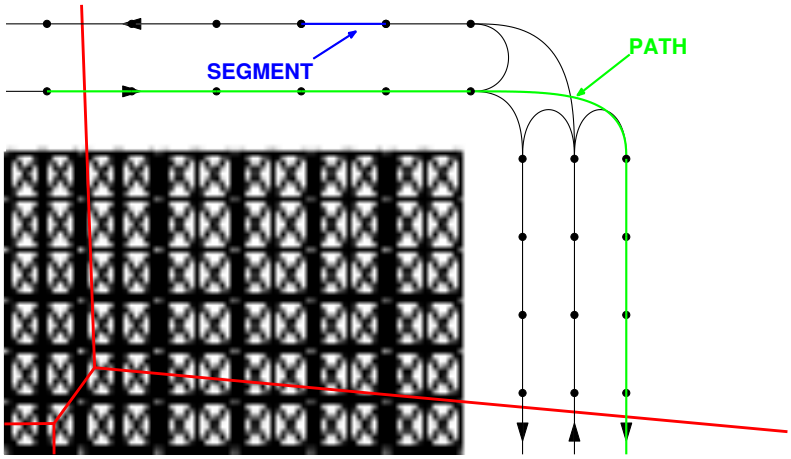


Figure 2.2: Definitions of Path and Segment

Definition 2.3. Strong Connectivity [41] *$\mathcal{G}$ is strongly connected when for all nodes $v_i, v_j \in V$, there exists a path in $\mathcal{G}$ between v_i, v_j.*

Definition 2.4. Redundancy *There exist multiple paths that connect each pair of nodes.*

It is worth noting that increasing the redundancy between each pair of nodes implies increasing the possibility of multiple choices of the paths. Thus, redundancy is an important index of quality to evaluate the roadmap for automatic warehouses. A roadmap with high redundancy globally entails a smaller number of stops of the AGVs due to traffic reasons, and a subsequent benefit for the efficiency of the system.

The objective of the algorithm is then to define a roadmap that fills the environment with feasible paths. Generally speaking, the free space $\mathcal{C}_{free}$ of an industrial environment is characterized by two main entities: *corridors* and *intersections*. Given a polygonal environment and its medial axis $\mathcal{Q}$, a corridor is formally expressed as follows.

Definition 2.5. Corridor *A corridor is the set of points in $\mathcal{C}_{free}$ bounded by at least two obstacles and such that $\mathcal{Q}$ is a segment of line parallel to the boundary of the obstacles.*

An intersection is a bounded portion of $\mathcal{C}_{free}$ in which more corridors flow. Formally:

Definition 2.6. Intersection *An intersection is the set of points in $\mathcal{C}_{free}$ bounded by corridors and whose medial axis $\mathcal{Q}$ is not parallel to the boundary of any of the obstacles.*

Fig. 2.3 shows a real warehouse and some intersections and corridors are underlined. Thus intersections and corridors provide a topological representation of the warehouse.

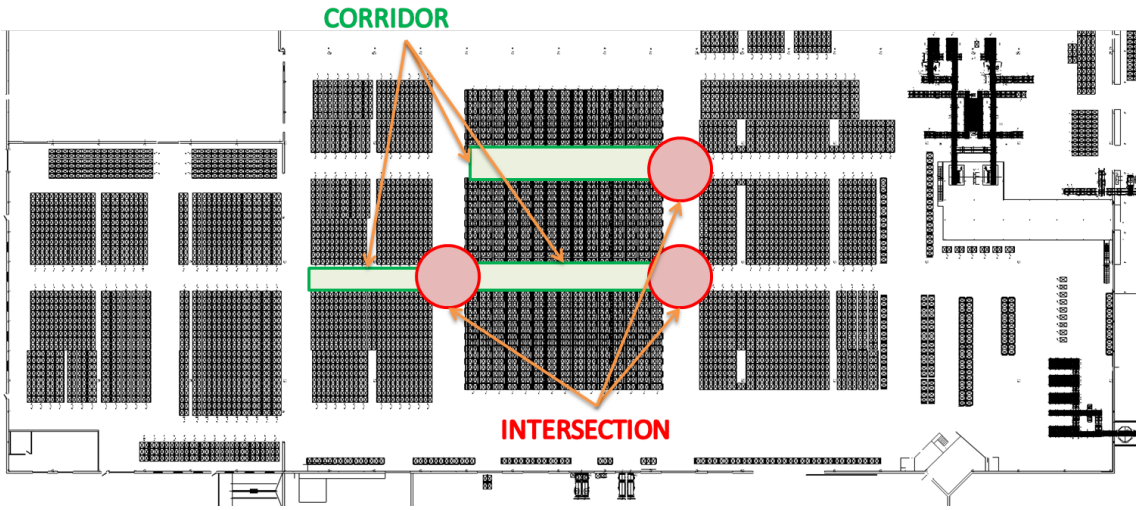


Figure 2.3: Intersections and corridors in an industrial environment

Furthermore some **Assumptions** are required:

A1 The free space in an intersection is large enough to contain at least one AGV.

A2 A corridor is longer than the dimension of one AGV.

The proposed method provides a roadmap which is feasible, optimal (according with an objective function), redundant and that covers all the free space.

2.3 The Algorithm

In this section, the algorithm for the generation of the roadmap will be detailed described. The roadmap generation algorithm is also reported in the author' papers [42, 43].

The layout of the industrial plant is provided as an input and it contains geometric information such as the position of the static obstacles in the environment. The proposed algorithm is composed of the following steps which will explained in the corresponding subsections:

A. First, the free space has to be covered.

B. Then the number of roads is maximized to fill this space.

C. A direction is subsequently assigned to each road in such a way that a specific parameter of the graph associated to the roadmap is maximized.

D. The roadmap is then smoothed considering the geometry of the AGV.

2.3.1 Find Corridors and Intersections

First of all, the free space has to be detected. Without loss of generality, assume that the obstacles are 2D polygons (e.g. bounding box). The free space can be captured by means of the medial axis transform (MAT), an example is shown in Fig. 2.4. In this way, the free space is modeled as a topological graph where each node represents an intersection and each edge a corridor, this is a *high clearance roadmap* which covers the space. The MAT process generates a set of segments of line. By analyzing the MAT it is possible to identify corridors and intersections, exploiting Definition 2.5 and Definition 2.6. Fig. 2.5 shows the intersections and corridors found in a portion of a real warehouse.

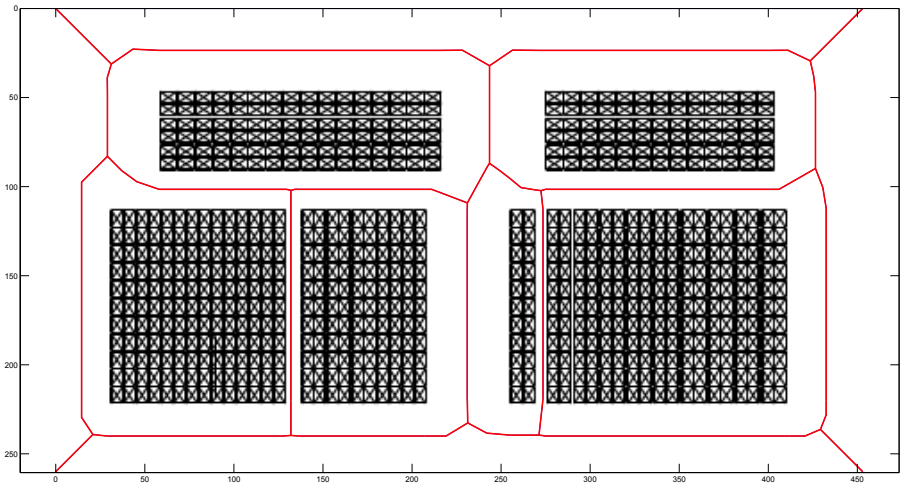


Figure 2.4: MAT

2.3.2 Fill the Corridors

The redundancy requirement is satisfied maximizing the number of roads. The roads have to be collision free as much as possible (collisions can take place in proximity of the intersections). Let δ be the maximum width of an AGV. Each road has to guarantee a minimal distance δ form the other roads in order to avoid collisions.

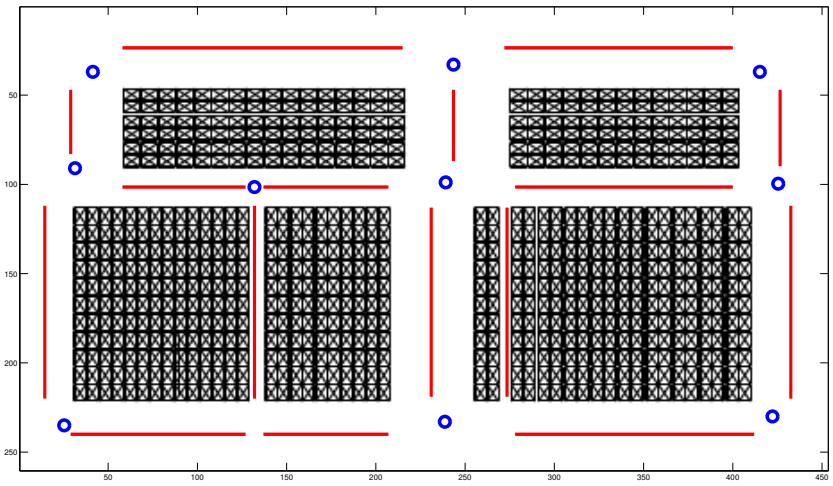


Figure 2.5: Find Corridors and Intersections: the circles identify the intersections and the straight lines identify the corridors.

This information is used to determine the maximum number of roads that guarantee at least the minimal distance δ from each other. Thus a set of roads is generated in each corridor, Fig. 2.6 shows an example. Practically, the width of each corridor is compared to the distance δ in order to obtain the maximum number of roads which can lay in the corridor. Subsequently the new roads are added to the roadmap. The filling procedure is described in details in Algorithm 1.

Algorithm 1: Fill the corridors

Data: Roadmap $\mathcal{G}$, corridors, AGV's dimension
Result: Roadmap with filled corridors

```
forall the corridor of C_free do
    calculate distance transform DT;
    compute maximum number of roads based on AGV's dimension;
    for i=1 to number of roads do
        generate road l ;
        put l in G ;
    end
end
```

2.3.3 Build the Intersections

In this section how to connect the roads of different corridors in an intersection is shown. At this stage, roads are straight lines. An intersection can be modeled as a

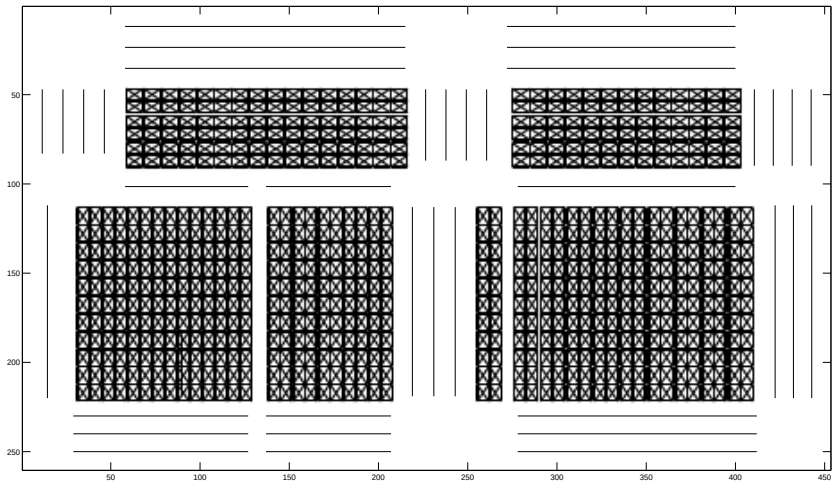


Figure 2.6: Fill the corridors: several roads are built in the corridors. The process is based on the dimension of the free space according with the dimension of the AGV.

polygonal area bounded by obstacles and corridors, as shown in Fig. 2.7a. First, the generation of a grid that extends all the roads is proposed, see Fig. 2.7b. In general, the extended roads can overlap each other and the grid is pruned considering the following constraints:

- the distance from obstacles has to be more than δ
- the distance form other roads has to be more than δ

In case that one of the constraints is not respected, the road is merged with another one or deleted, see Fig. 2.7c.

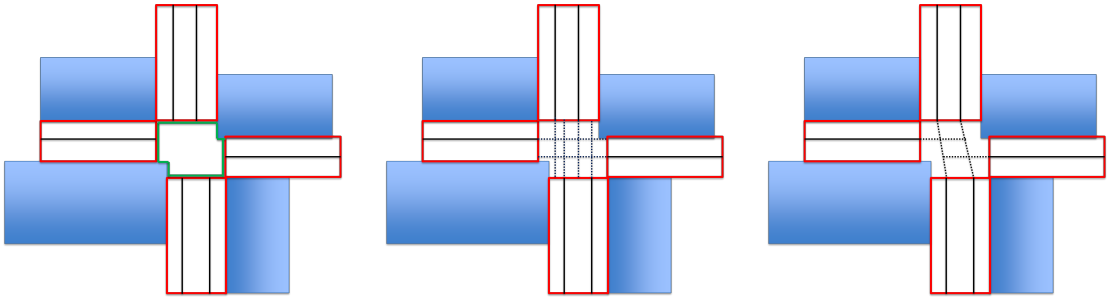
The result of this process applied on a portion of a real environment is shown in Fig. 2.8 and the details of the algorithm are described in Algorithm 2.

2.3.4 Assign Directions

This step aims at assigning the direction to each road maximizing a general objective function. Two objective functions have been used:

- the algebraic connectivity
- the maximum flow

The environment is topologically described as a weighted directed graph $\mathcal{T}(\overline{V}, \overline{E})$ with weight function $\Omega : \overline{E} \rightarrow \mathbb{R}$. The vertices $\overline{V}$ model the intersections and the



(a) Intersection (b) Extend the roads (c) Final Intersection

Figure 2.7: The intersections are created merging or deleting roads form different corridors. The intersection region is outlined with a green polygon and the corridors with a red one. The roads in the corridors are shown as black lines. This process is outlined in Algorithm 2

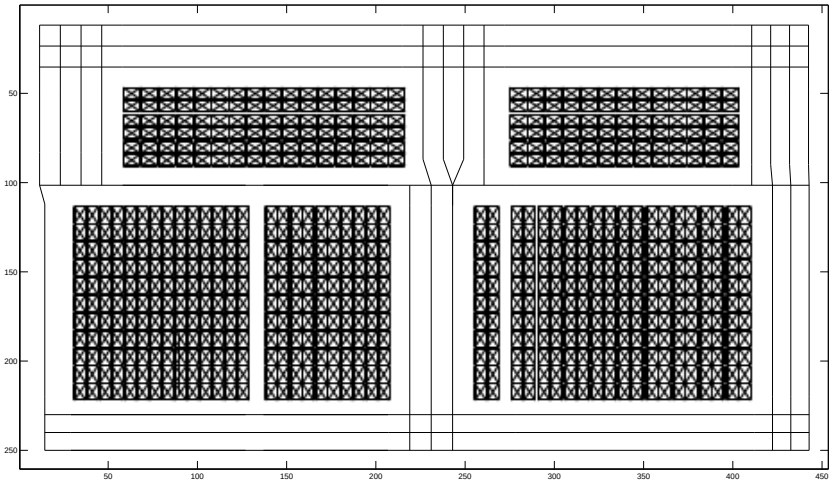


Figure 2.8: Build the Intersections: the intersections are created among roads from different corridors.

Algorithm 2: Build the Intersections

```
Data: Roadmap 𝒢 , intersection
Result: Roadmap
forall the intersection do
    find the coming corridors C_in ;
    initialize grid intersect_grid ;
    forall the C_in do
        forall the roads in C_in do
            extend the road until the next corridor ;
            put new extended line in intersect_grid ;
        end
    end
    forall the i = 1 to lines in intersect_grid do
        forall the j > i to lines in intersect_grid do
            if lines_i is directed to obstacles then
                delete lines_i ;
                break ;
            end
            if lines_j is directed to obstacles then
                delete lines_j ;
                continue ;
            end
            if distance_{i,j} < distance_min then
                merge lines_i and lines_j;
            end
        end
    end
end
```

edges $\overline{E}$ model the corridors among the intersections. Each corridor can be represented by one or two edges, based on the number of roads obtained after applying Algorithm 1. The weight of each edge in a corridor is a function of the number of roads and of the length of the corridor itself. In particular, the traffic flow of the AGVs among the intersections is modeled as the communication among nodes interconnected with the graph $\mathcal{T}$. The communication efficiency on a graph increases as the edge weights increase. Therefore, for each corridor, the corresponding edge weight are defined as quantity that is:

- directly proportional to the number of roads (more roads allow more AGVs to simultaneously travel on the corridor)
- inversely proportional to the length of the corridor (long corridors require more

time to be traversed)

Formally the weight of the edge between the i-th and j-th intersection can be expressed as follows.

$$w_{i,j} = \frac{n_{roads,i\to j}}{\mathfrak{L}_{i,j}} \in \Omega \tag{2.1}$$

Where $\mathfrak{L}_{i,j}$ is the Euclidean distance between the intersection i and j and $n_{roads,i\to j}$ is the number of roads directed from i to j. The direction of each road is obtained as a solution of the *Direction Assignment Process*, formalized as follows.

Definition 2.7. Direction Assignment Process *A direction has to be assigned to each road in the roadmap $\mathcal{G}$ in such a way that the topological graph $\mathcal{T}$ is strongly connected and a defined objective function is optimized.*

The corridors can be divided into two classes: bi-directional corridors and mono-directional corridors. Corridors with more than one road are defined as **bi-directional corridors** and are represented by two edges. A corridor with an even number of roads from i to j is characterized by $n_{roads,i\to j} = n_{roads,j\to i}$ which implies $w_{i,j} = w_{j,i}$. Vice-versa a corridor with an odd number of roads from i to j is characterized by $n_{roads,i\to j} = n_{roads,j\to i} \pm 1$ which implies $w_{i,j} \neq w_{j,i}$. The exact values of $n_{roads,i\to j}$ and $n_{roads,j\to i}$ are computed solving the Direction Assignment Process.

Corridors with only one road are defined as **mono-directional corridors** and are represented by one edge, whose direction is computed solving the Direction Assignment Process. If the edge is directed form i to j, its weight is computed as $w_{i,j} = 1/\mathfrak{L}_{i,j}$.

The Direction Assignment Process is an optimization problem that will be solved in the following steps:

1. Randomized direction assignment
2. Strong connectivity verification
3. Optimization and connectivity evaluation

Randomized direction assignment

Define n_k as the number of roads obtained in the k-th corridor after applying Algorithm 1. Assume, without loss of generality, that the k-th corridor connects the i-th and j-th intersections. The directions of the roads are assigned consistently with the previously introduced definitions of bi-directional and mono-directional corridors. This can be obtained as follows:

- $\left\lfloor \frac{n_k}{2} \right\rfloor$ roads with direction $i \to j$
- $\left\lfloor \frac{n_k}{2} \right\rfloor$ roads with direction $j \to i$
- $n_k - 2 \cdot \left\lfloor \frac{n_k}{2} \right\rfloor \in \{0, 1\}$ roads with random direction

Summarizing, only the road of each mono-directional corridor and one road of each odd bi-directional corridor have to be defined. In this way the problem of assigning the directions is drastically reduced.

Strong connectivity verification

In this step, we introduce a procedure to verify that a particular solution of the randomized direction assignment described in Section 2.3.4 generates a strongly connected roadmap.

As described in Section A, the algebraic connectivity is an indicator of strong connectivity for balanced graphs. It is worth noting that, in general, $\mathcal{T}$ is not balanced. However a balanced version of the graph $\mathcal{T}$ can always be obtained by means of opportunely computed weights of the edges $\mathbf{p} = [\mathbf{p}_1, \cdots, \mathbf{p}_M]$ where M is the number of the edges [44]. The weight vector that balances a directed graph is the solution of $\mathcal{I} \cdot \mathbf{p} = \mathbf{0}$, where $\mathcal{I}$ is Incidence matrix of $\mathcal{T}$ and $\mathbf{0}$ is the null vector. The graph $\overline{\mathcal{T}}$ is the balanced version of the graph $\mathcal{T}$, and according to Eq. (A.3), its Laplacian matrix is defined as follows:

$$\mathcal{L}_{\mathbf{p}} = \mathcal{I} \cdot diag(\mathbf{p}) \cdot \mathcal{I}^T \tag{2.2}$$

In other words, $\overline{\mathcal{T}}$ is a balanced graph with the edges directed as those of $\mathcal{T}$. Therefore, $\overline{\mathcal{T}}$ is strongly connected if and only if $\mathcal{T}$ is strongly connected as well.

The algebraic connectivity can then be used to verify the strong connectivity of $\overline{\mathcal{T}}$. In particular if $\Re\left\{\lambda_2^{\overline{\mathcal{T}}}\right\} > 0$ then the previously assigned directions of the roads generate a strongly connected roadmap, that represents an admissible solution.

Optimization and connectivity evaluation

In general, different solutions for the randomized direction assignment may exist that lead to a strongly connected graph. In order to compare these solutions, different parameters can be selected. In this work, we focused on the algebraic connectivity of $\mathcal{T}$ and on the maximum flow of $\mathcal{T}$ as indexes to be maximized. As described in

Section A, increasing the algebraic connectivity improves the overall efficiency of the graph. Instead, preferred directions can be maximized by maximizing the maximum flow between two locations.

The overall Direction Assignment Process can now be modeled as a binary non-linear optimization problem. The optimization parameters assume only two scalar values (0,1) and they represent the directions defined in the randomized direction assignment described in Section 2.3.4. The optimization variables can then be collected in the vector x, defined as

$$x = [d_1 \cdots d_{\overline{n}}]^T \in \{0\ 1\}^{\overline{n}}$$

where $\overline{n}$ is the number of roads to which a direction has to be assigned. The optimization problem is formalized as:

$$\begin{aligned} \text{maximize} \quad & g(\mathcal{T}, x) && \text{(2.3a)} \\ \text{subject to} \quad & x \in \{0\ 1\}^{\overline{n}} && \text{(2.3b)} \\ \text{and} \quad & \Re\{\lambda_2^{\overline{\mathcal{T}}}(x)\} > 0 && \text{(2.3c)} \end{aligned}$$

The constraint defined by the Eq. (2.3c) represents the strong connectivity verification described in Section 2.3.4. The objective function shown in Eq. (2.3b) represents the parameter of the graph $\mathcal{T}$ to be maximized. In general, the algebraic connectivity is preferred as a heuristic to maximize the traffic flow since it is a global parameter of the roadmap. The maximization of the algebraic connectivity consists of maximizing the real part of the second smallest eigenvalue of the Laplacian matrix associated with the strongly connected graph $\mathcal{T}$. The objective function for the algebraic connectivity maximization is then given by:

$$\text{maximize} \quad \Re\{\lambda_2^{\mathcal{T}}(x)\} \tag{2.4}$$

The maximum flow optimization is the other considered heuristic. It is formally described by the pair origin-destination (o, d), whose elements are nodes of the roadmap. The road directions are assigned by maximizing the maximum flow among the pair (o, d). Let $F(o, d)$ be the maximum flow between the given pair (o, d), the new objective function is formalized as:

$$\text{maximize} \quad F(o, d) \tag{2.5}$$

As noted the optimization problem in Eq. (2.3) is generally NP-hard. However there are several algorithms that can generate a good approximate solution to this

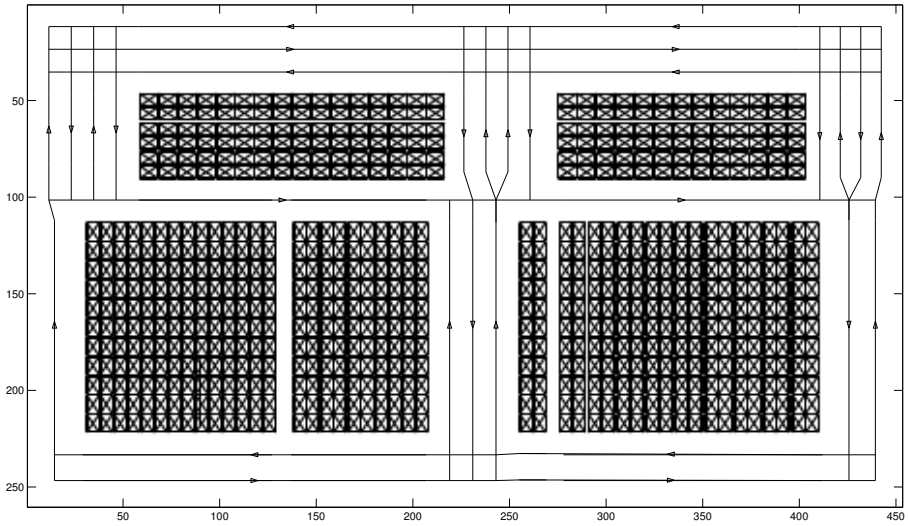


Figure 2.9: Assign Directions: the triangles represent the directions of the roads.

class of problems for an effort that increases only slowly as a function of $\overline{n}$. In our approach a local search method is used. In particular, a *Tabu Search* approach is combined with a *Monte Carlo* method [45].

An example of solution for the *Direction Assignment Process* is shown in Fig. 2.9.

2.3.5 Smooth the Roadmap

The roadmap has to be smoothed in order to generate feasible trajectories with respect to the AGV's kinematics and geometry. The previous steps of the algorithm build a roadmap composed by straight lines. Naturally these lines intersect each other forming sharp corners. This method consists of substituting the sharp corners with *Bezier curves* [46] which are built with respect to the minimal radius of curvature of the AGVs. In this way the roadmap is composed by a sequence of segments of line and *Bezier curves*. In particular, the curves are used to connect the roads which are intersecting. The result of the smoothed and final roadmap is visible in Fig. 2.10

2.4 Experimental Validation

The proposed methodology was evaluated on real industrial plants comparing the obtained results with currently utilized roadmaps. It is worth noting that current

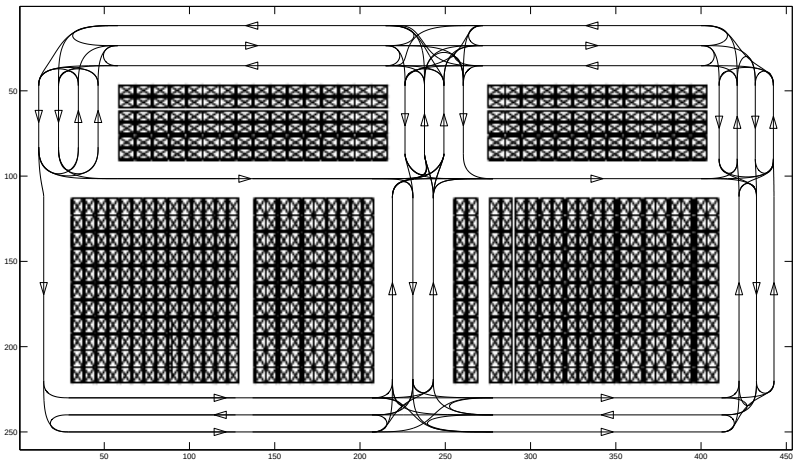


Figure 2.10: Smoothed and final roadmap

industrial state of the art techniques are based on manually built roadmaps. Hereafter it is possible to refer to the roadmaps generated by the proposed method as *automatic roadmaps*, and to the roadmaps currently used in industrial warehouses as *manual roadmaps*. The comparison is performed by analyzing the connectivity of the roadmaps and the redundancy. The algebraic connectivity is then used as a heuristic for evaluating the quality of a roadmap and the maximum flow [41] is used as index of redundancy. The higher the algebraic connectivity, the better is the connectivity of the roadmap, and the higher the maximum flow the better is the capability of the roadmap to guide all the AGVs towards their destination with no congestions due to waiting. The comparison was performed on plants with different dimensions. Due to confidentiality reasons, the manual roadmaps cannot be shown or discussed in detail. In general the plants can be divided into three classes based on the dimension. Namely:

- *small*: less than 10 AGVs
- *medium*: from 10 to 30 AGVs
- *big*: more than 30 AGVs

Table 2.1 shows the results obtained on three typical plants representing the above mentioned classes.

It is worth noting that the proposed roadmaps are comparable with the manual ones in terms of connectivity and redundancy of the graph. Observing the algebraic

Class	**Dimension** (AGVs)	**Manual Roadmap** $\Re\{\lambda_2^{\mathcal{T}}\}$	*max flow*	**Automatic Roadmap** $\Re\{\lambda_2^{\mathcal{T}}\}$	*max flow*	**Max.Flow increase**
small	5	0.0031	35.1	0.0043	40.4	15%
medium	25	0.0002	59.2	0.0027	181.2	306%
big	50	0.00075	29.6	0.0019	212.5	718%

Table 2.1: Values of connectivity and maximum flow

connectivity term (real part of the second smallest eigenvalue of the Laplacian matrix), it is possible to state that the connectivity of the automatic roadmaps is higher than the connectivity achieved by the manual ones. The automatic roadmaps are also more redundant (more paths) than the manual ones. Figs. 2.12, 2.13, 2.14, 2.15 show the automatic roadmaps generated with the proposed approach.

The roadmap has been also evaluated in terms of different algebraic connectivity values. In this case, simulations are performed by considering a fleet of AGVs moving along the roadmap. The term of comparison is the total crossing time and the variables are the algebraic connectivity of the roadmaps and the number of AGVs. This allows us to evaluate the choice of the algebraic connectivity as a heuristic. The following Table 2.2 shows the values of algebraic connectivity for different roadmaps. Hereafter we will refer to the layouts with their identification number.

Layout	**Maximized** λ_2	**Not-maximized** λ_2
1	0.0063	0.0037
2	0.0043	0.0024
3	0.0027	0.00098
4	0.0019	0.00083

Table 2.2: Values of algebraic connectivity (λ_2) used in simulations

The results (see Fig. 2.11) show that the algebraic connectivity actually affects the performance of the system, estimated by means of the total crossing time: as expected, in each scenario, an optimized value of the algebraic connectivity generates better results. Namely, the higher is the algebraic connectivity, the lower is the crossing time.

2.5 Conclusion

In this chapter an automatic roadmap generation process that autonomously defines a set of near-optimal paths in order to cover and connect all the free space has been

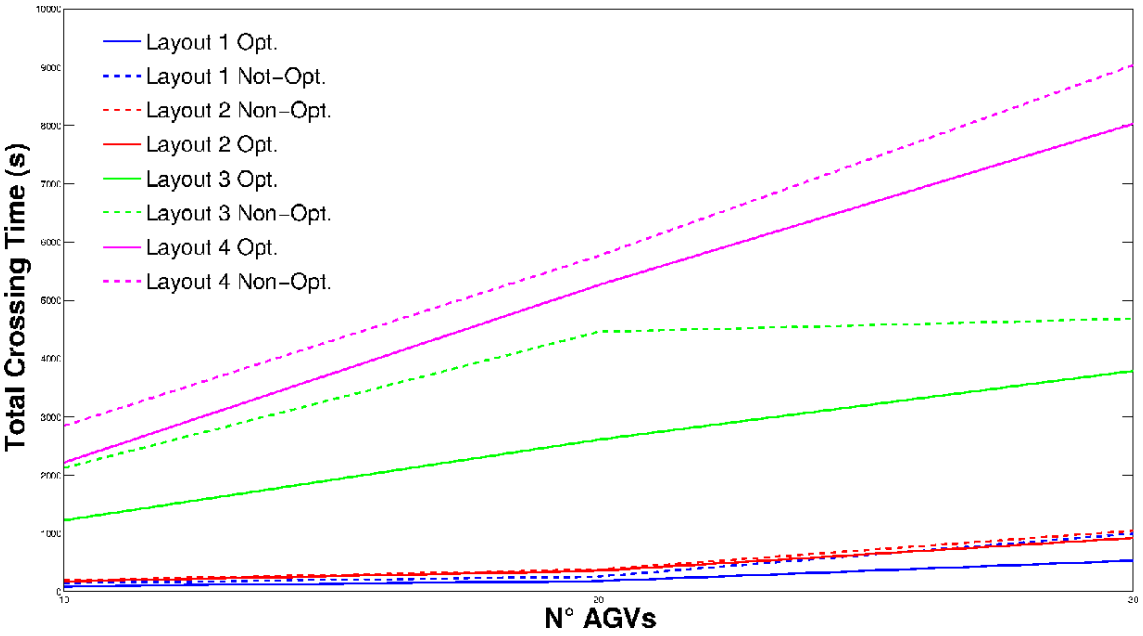


Figure 2.11: Algebraic Connectivity Validation

described.

An algorithm is proposed to build a roadmap in such a way that the coverage, the redundancy and the connectivity are maximized. A roadmap has to cover all the free space in order to reach all the positions of interest, that is the operation points. The coverage is guaranteed by means of the medial axis transform (MAT) of the free space. The redundancy is an important index of quality to evaluate the roadmap of industrial environments. A roadmap with high redundancy entails a decrease of the down time of the overall system due to traffic reasons. The redundancy is achieved by maximizing the number of roads and paths. The connectivity of the roadmap affects the efficiency of the associated graph. The algebraic connectivity is then chosen as a heuristic to maximize the traffic flow. Naturally the roadmaps have to be strongly connected. The simulations have shown that it is possible to generate an automatic roadmap with higher connectivity than a manual roadmap built on the same real plant of an industrial warehouse.

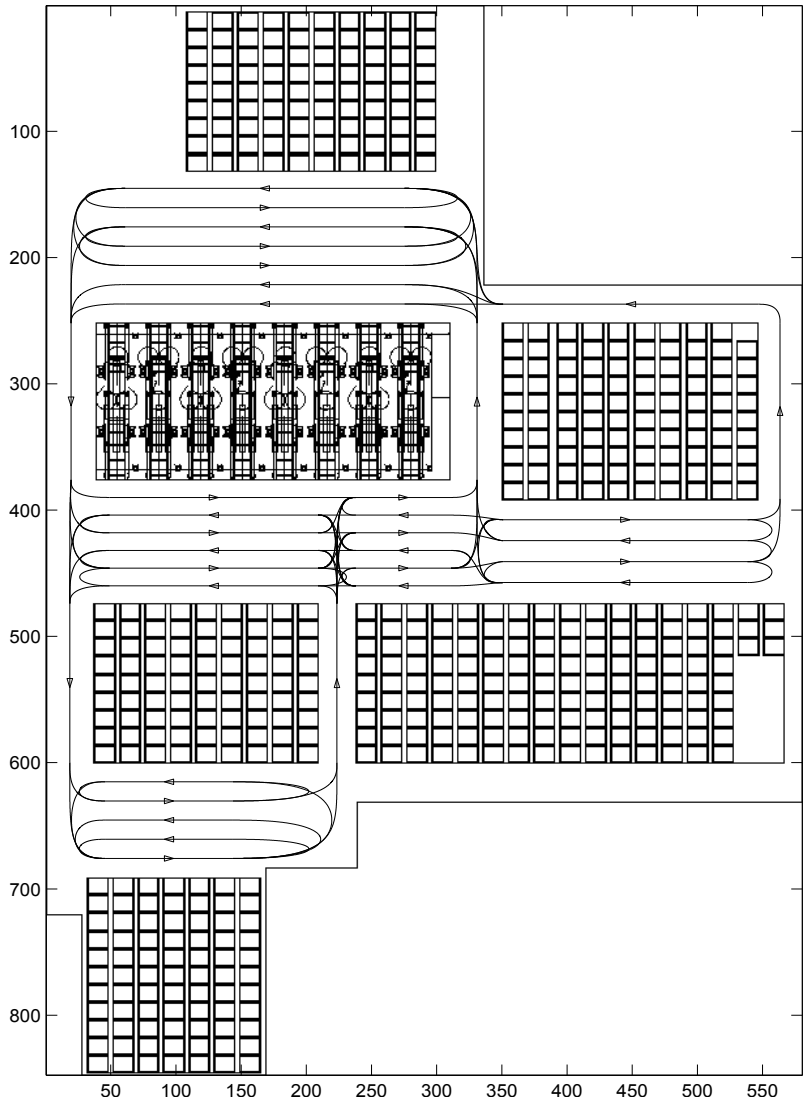


Figure 2.12: Automatic Roadmap of a typical *small* industrial warehouse

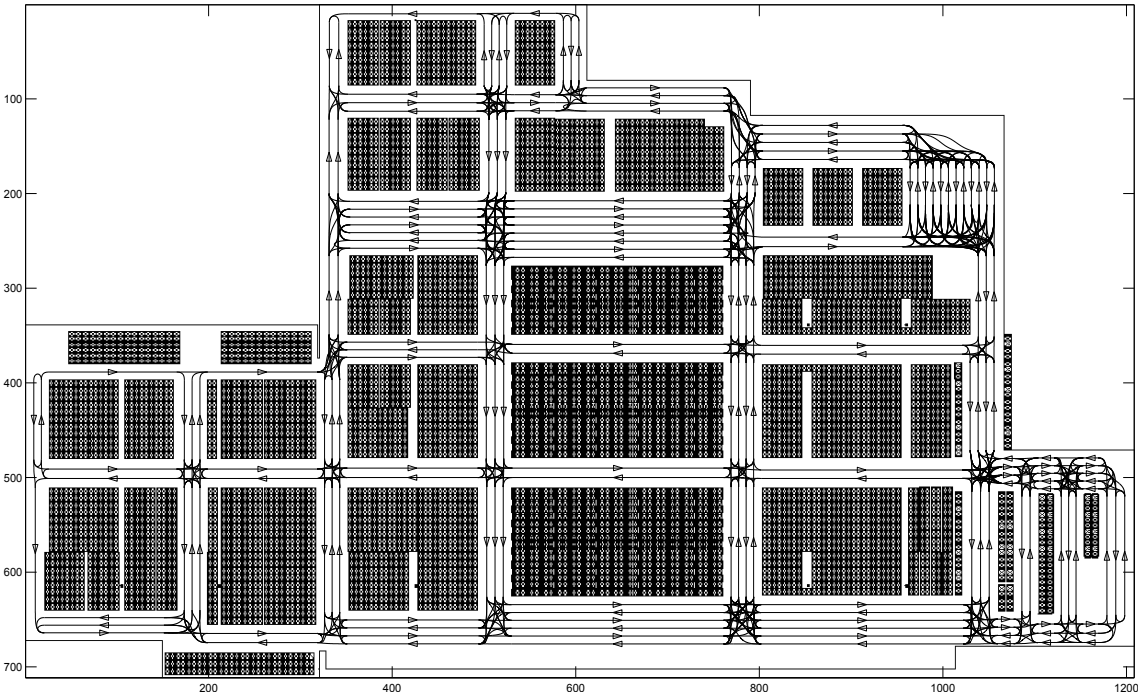


Figure 2.13: Automatic Roadmap of a typical *medium* industrial warehouse

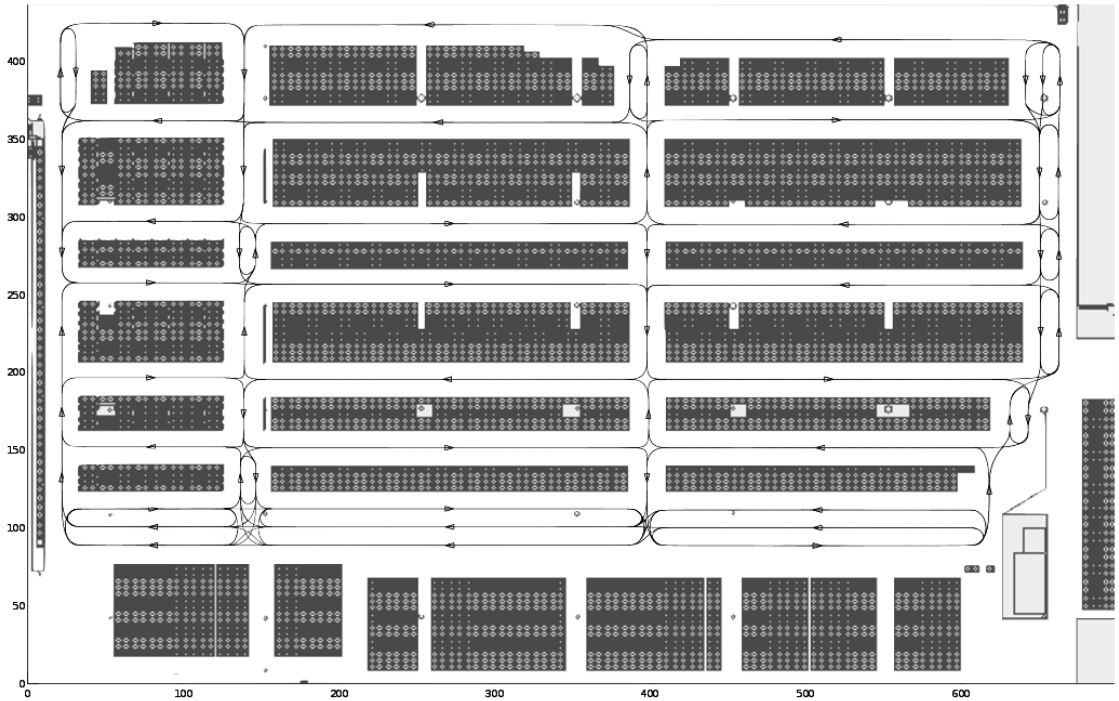


Figure 2.14: Automatic Roadmap of a typical *medial* industrial warehouse

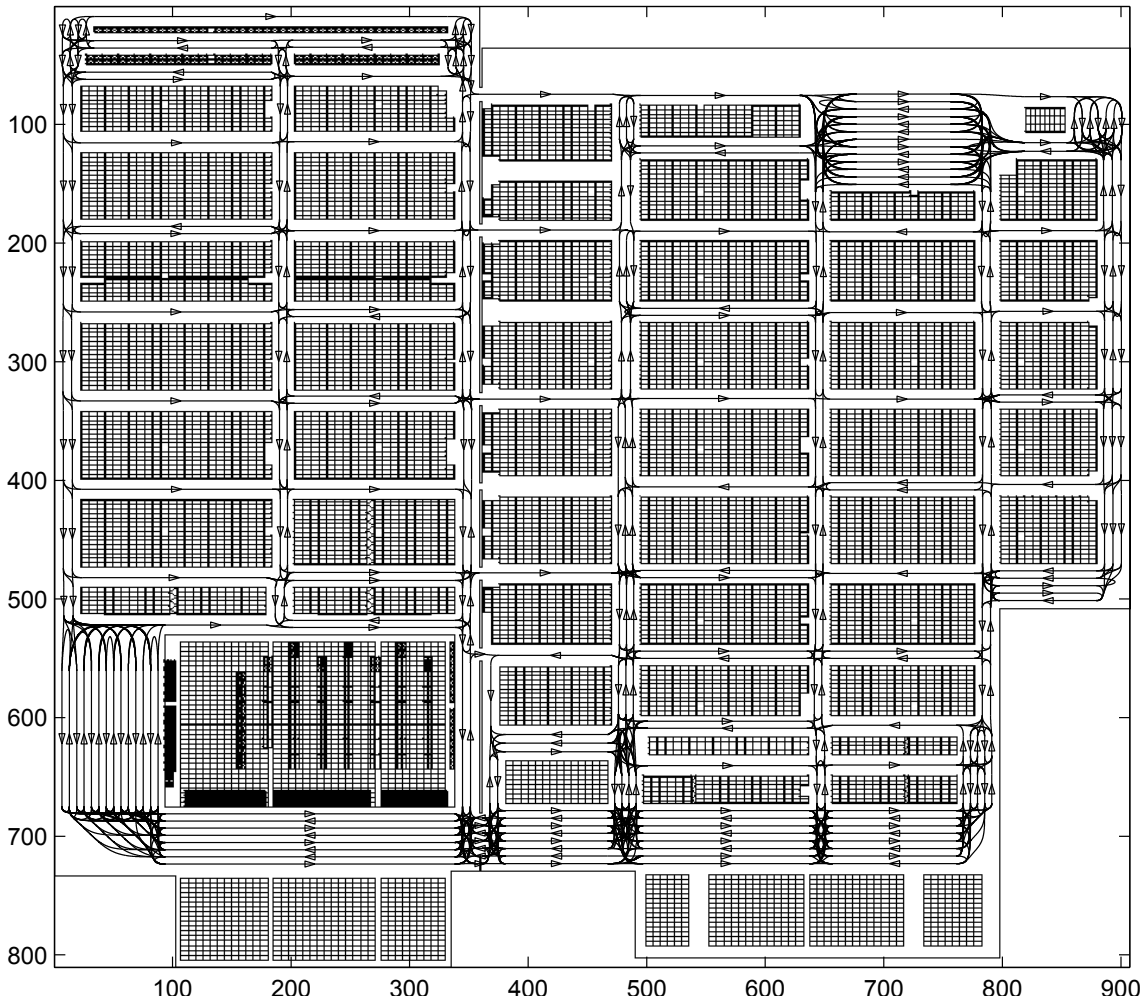


Figure 2.15: Automatic Roadmap of a typical *big* industrial warehouse

Chapter 3

Hierarchical Control Architecture

In this chapter, an architecture for the coordination of a fleet of Automated Guided Vehicles (AGVs) is proposed. The architecture is based on a hierarchical structure composed of two levels of abstraction, in this way the complexity of the coordination problem can be reduced.

3.1 Introduction

Coordinating a fleet of AGVs (or robots) can be a complex task. Most of the solutions presented in literature, the coordination focuses only on single aspects of the multi-robot system. For instance, some solutions propose navigation algorithms, others path planning techniques or roadmap-based solutions. However the system is not considered as a whole in such approaches. In fact, optimizing a local cost function is not sufficient for maximizing the global performance. A significant example of this claim can be found in the fact that often, in current industrial application, manual traffic rules are used to overcome local issues due to the failed matching between the design of the roadmap and the coordination algorithm [5]. It is worth noting that the performance of the coordination algorithm is strongly related to the characteristics of the roadmap where the coordination itself is performed. Hence, even though the design of the roadmap and the subsequent coordination are strongly related problems, in the literature they are generally treated in a separate manner, to the best of the authors' knowledge. In fact, while several strategies can be found in the literature for the definition of the roadmap, none of them considers the subsequent coordination that will be perform along the roadmap itself, during the daily industrial operations. Thus, the main idea of this thesis is to consider the problem as a whole, rather than independently finding optimal solutions for parts of it.

Our approach aims at solving the problem of coordination in a holistic way: in

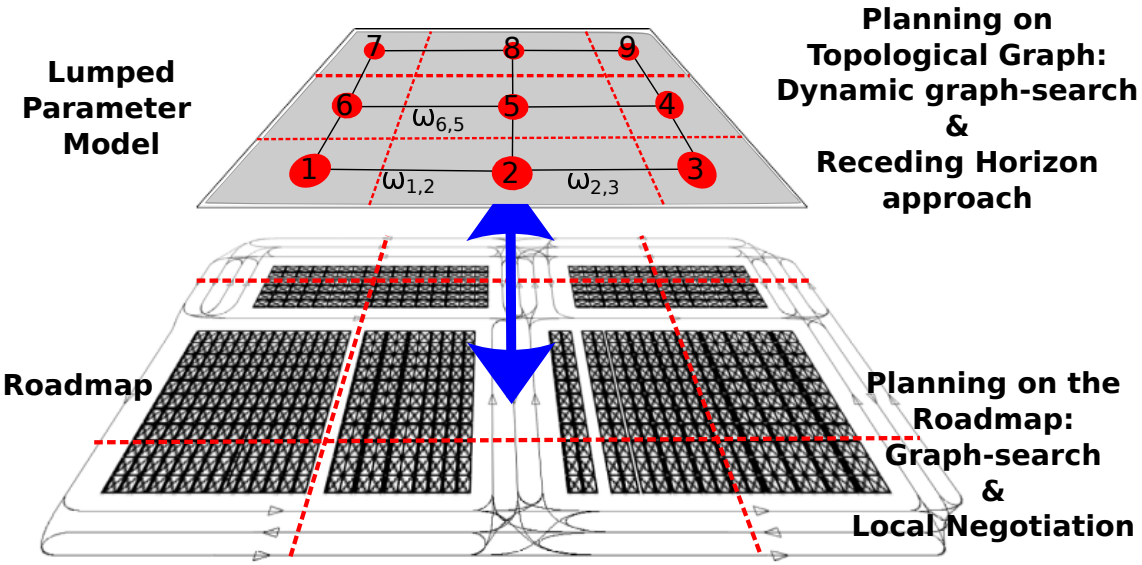


Figure 3.1: System Architecture Overview

this chapter an ensemble approach where a two layer control architecture is proposed. In addition, the automatic algorithm for the definition of the roadmap described in Chapter 2 is extended in order to be merged with the hierarchical architecture.

The dimension of the multi-robot space is reduced using of a multi-layer structure to represent the world. As explained in [47], the approach is to construct a hierarchical map which can abstract the traversable areas using the adequate number of nodes and edges of a graph. The path is searched using the graphs of the several layers. In our solution, two layers (*hierarchical planner*) are used. In the low-level, the structure of the generated roadmap is considered and a *lumped parameter model* representing the traffic is built. The high-level abstracts the relationship among different macro-areas of the environment, defining a topological map. On both these levels a coordination algorithm is performed. Furthermore, the coordination is obtained as a combination of centralized and decentralized approaches. In particular, a partially decentralized coordination is proposed which exploits shared resources (i.e. centralized information) and local negotiation (i.e. decentralized coordination). Practically, the coordination is faced by using a model predictive control approach together with graph search methods and with local negotiations based on priorities. Fig. 3.1 shows the overall system architecture.

3.1.1 Outline

The rest of the Chapter is organized as follows. Section 3.2 describes the problem,

3.2 Problem Statement

In this Chapter we will present a strategy to solve the problem of coordinating a fleet of AGVs in an industrial environment. The objective is to maximize the performance of the overall system. A hierarchical path planning method is then adopted on a roadmap. The roadmap can be generated by different methods, for instance with the method presented in Chapter 2.

A *hierarchical planner* is then proposed in order to coordinate a fleet of AGVs on a predefined roadmap. As explained in Chapter 2, a roadmap is a set of routes, composed of distinguished elements called segments (see Fig. 2.2). The AGVs are constrained to follow the roadmap and its segments. In particular, each segment can be allocated only to one AGV at time and it is characterized by a unique direction of movement. These properties are useful for coordination purposes, since it is not possible to have pathological situations, such as two (or more) AGVs moving along the same segment with opposite directions or conflicts on the same segment. A path is a set of feasible segments from a source position to a target destination. A path can be assigned to an AGV, that is then allowed to move along the segments in the path.

We introduce the following definition of *admissible set of paths*.

Definition 3.1. Admissible set of paths *Given a fleet of $\mathcal{N}$ AGVs, a set of $\mathcal{N}$ paths is* admissible *if, assigning each path to a different AGV, a velocity profile can be defined in such a way that collisions are avoided.*

The problem can be formally stated as follows:

Problem 3.1. Multi AGV Coordination *Consider:*

- *a fleet of $\mathcal{N}$ AGVs*
- *the initial and final positions for all the AGVs*
- *the map of the elements in the environment*

Define a coordination strategy such that each AGV is able to move from its initial position to its assigned final position minimizing the total crossing time and by avoiding conflicts.

Therefore, the problem consists of planning a path for a fleet of AGVs in a 2D static environment, so that conflicts and deadlocks are avoided. Each AGV starts its path in an initial position, and has to reach its own final position. Each AGV

can communicate with the others in its neighborhood and has a prior knowledge of the environment.

The following **Assumptions** are made on the system:

A1 No unforeseen events, such as the presence of dynamic obstacles (manual forklift, people, etc.) are considered.

A2 Each AGV has prior knowledge of the roadmap.

In the next sections we will provide a solution to Problem 3.1: in particular, we will first extend the automatic algorithm for the creation of a roadmap, along which the motion of the AGVs will be subsequently coordinated.

The task assignment (that is the assignment of the initial and final position for each AGV) is out of the scope of this chapter, and therefore it will be not considered.

3.3 Roadmap Model for Coordination

In this section an extension of the roadmap generation algorithm is provided. The additional features are required in order to coordinate the fleet of AGVs with the method which will be described in the rest of the chapter.

Given a roadmap that covers the free space, it is useful to cluster the segments into topological entities, that will be hereafter referred to as **sectors**. Specifically, a sector S is an area, or a region, which can be distinguished from the other ones based on:

- topological aspects
- logistical aspects
- geometrical aspects
- particular constraints.

The division in sectors gives a topological representation of the roadmap, that provides information regarding the connection among the sectors. Furthermore, the sector division is built in order to bound the areas of traffic congestion in specific regions. In this way, a *lumped parameter model* of the traffic is built. In detail, a sector is defined in such a way that a connection with neighboring sectors exists. The constraints are defined based on the characteristics of the operational environment. For instance, constraints can be defined in terms of maximum number of AGVs contained in a single sector, or in terms of maximum number of operations of

loading/unloading. This kind of information is owned by the sectors and it is stored in a *centralized manner*. In this way, the information is visible to all the AGVs and is shared among them from the centralized storage. The sector division is performed by using the *Voronoi decomposition* whose seeds are located in the barycentre of the intersection areas. Fig. 3.2 shows an example of that decomposition in real industrial environment. The roadmap can then be represented, from a topological point

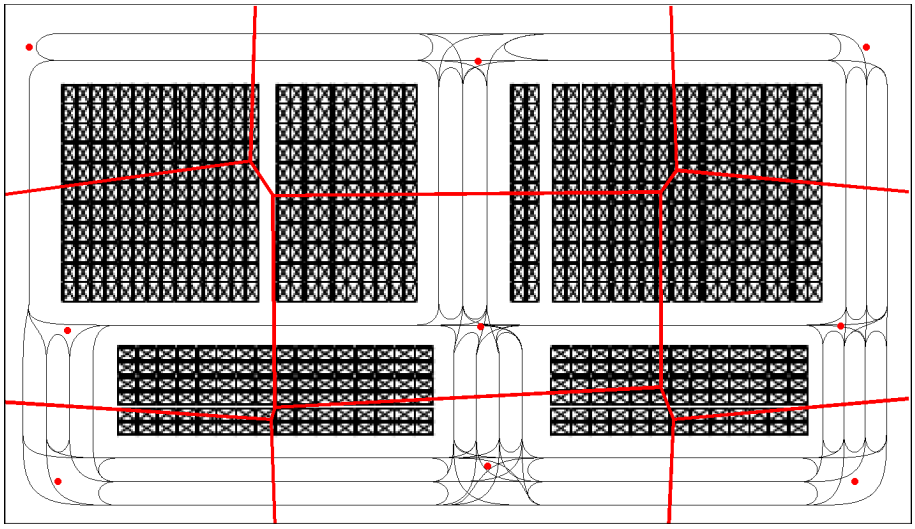


Figure 3.2: Sector division exploiting Voronoi decomposition

of view, by means of a directed graph $\mathcal{Z}$: each node of the graph is a sector, and each edge is a connection between two sectors. In particular, two sectors are connected if there exists a path on the roadmap which connects them and if the path is contained only within those sectors. The weight of the edge directed from the sector i to the sector j is given by:

$$\omega_{i,j} = K \cdot \frac{N_j}{C_j - N_j} + E_{i,j} \tag{3.1}$$

where i and j are two neighboring sectors, N_j is the current number of AGVs within the sector j, C_j is the maximum number of AGVs that the sector j can contain (that is the capacity), $E_{i,j}$ is the Euclidean distance among the centers of the two sectors and K is a static gain.

The automatic roadmap method concentrates all the crossroads in specific areas. In this way the congestion of the traffic is limited and it can be enclosed only to some specific areas. Let us define these areas as *intersection areas*. The intersection area is then the bounded region of a sector where the coordination problem occurs.

Let us define the q-th sector as $\mathcal{S}_q$ which contains $\mathcal{Z}_q$ paths, that is

$$\mathcal{S}_q = \{\pi^1, \ldots, \pi^{\mathcal{Z}_q}\}.$$

The i-th path within $\mathcal{S}_q$ is split into $\mathcal{M}_i$ entities called segments, namely:

$$\pi^i = \{\pi^i_1, \cdots, \pi^i_{\mathcal{M}_i}\},$$

as pictorially depicted by Fig. 2.2.

Let $\mathcal{V}(\pi^i_h)$ be the portion of space occupied by a vehicle moving along the segment π^i_h, that is the trace of the vehicle along that segment. A segment collides with another segment when their traces are intersecting. An intersection area $\mathcal{A}_q$ of $\mathcal{S}_q$ is then formally defined as:

Definition 3.2. ***Intersection Area*** $\mathcal{A}_q = \{\pi^l_h \mid \pi^l \in \mathcal{S}_q,\ \exists j = 1, \ldots, \mathcal{Z}_q,\ j \neq l,$ *and* $\exists r = 1, \ldots, \mathcal{M}_j$ *such that* $\mathcal{V}(\pi^l_h) \cap \mathcal{V}(\pi^j_r) \neq 0\}$

An intersection area is then the set of the colliding segments of different paths in the sector and, in general, each sector contains at least one intersection area.

When more AGVs are moving on the same intersection area, they have to avoid collisions among each other. For this reason crossroads, and thus the intersection areas, are considered as shared resources, to be allocated to a single AGV when necessary.An intersection area can be described also by using different types of segments (see Fig. 3.3), namely:

- *main segment*: segment on which the AGV has to go
- *intersecting segment*: segments intersecting the main segment
- *near segment*: segments that do not guarantee the security distance δ

As shown in Fig. 3.4, each sector is characterized by the presence of several *intersection areas* (the figure shows only one) and a certain number of *attention points*. These points are defined as:

Definition 3.3. Attention point *For each segment entering the intersection area, an attention point is defined in such a way that, if an AGV stops on the attention point itself, it does not collide with AGVs in the intersection area, in the sector, or in other attention points.*

It is worth noting that the definition of the attention points is based on the size and shape of the AGVs used in the particular application.

The automatic roadmap method provides then a *lumped parameter model* of the traffic: all the congestions are enclosed into the intersection areas. Based on this idea, the following coordination strategy is proposed.

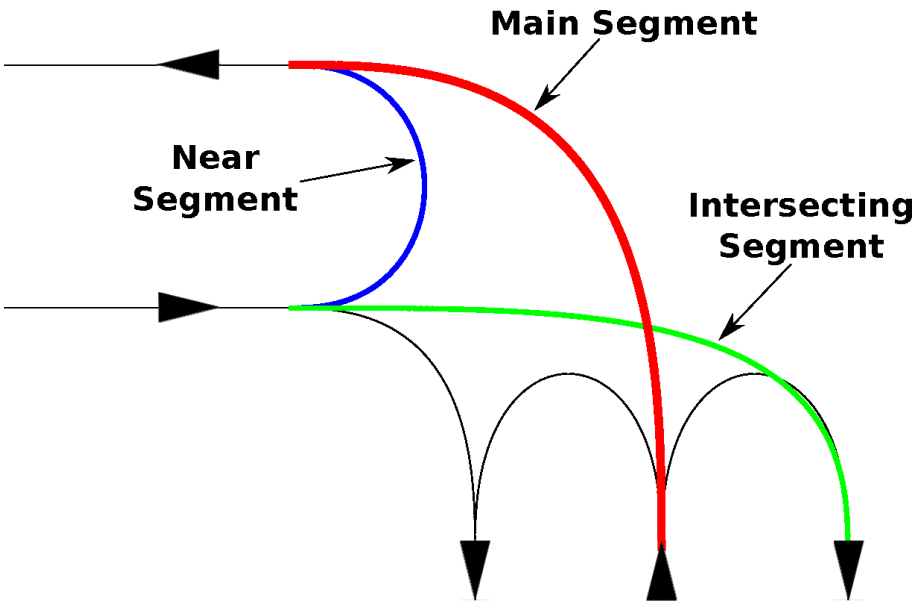


Figure 3.3: Roadmap model: definition of intersection area

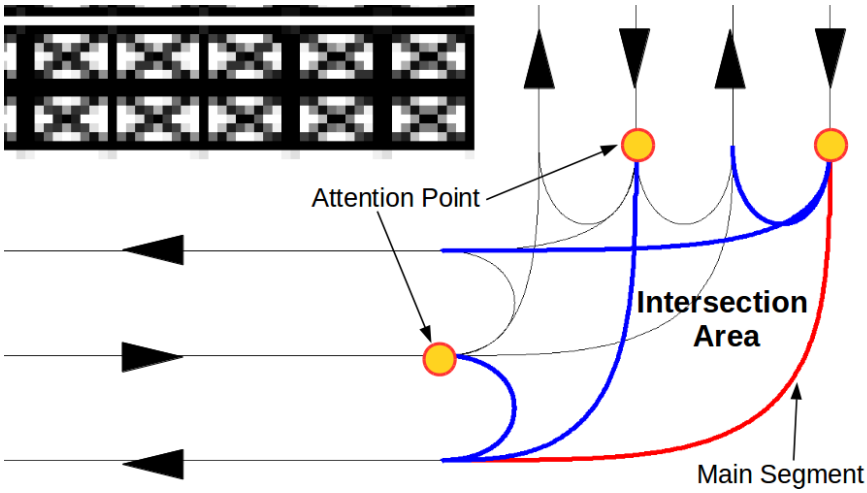


Figure 3.4: Roadmap model: example of intersection area

3.4 Two Layer Control Architecture

Since the roadmap is built in order to model the traffic as a lumped parameter model, where all the congestion areas are locally bounded and concentrated in specific points (Section 3.3), a multi-layer architecture for the coordination of the fleet of AGVs can be used. The proposed planning method is then based on a discretization of the environment and on concentrated traffic proprieties. In particular, in our approach, two layers are used. The top-layer, or *Topological Layer*, is a topological map representing the global map, composed of all the sectors. The layer below, or second layer, is the geometric map of each sector of the first layer, and will be

hereafter referred to as *Roadmap Layer*.

Therefore the path planning is done on two levels, thus defining a *hierarchical planner*. Topology path planning searches for the best path to the final goal (actually to the final sector where the real goal is) from the current sector. Roadmap planning computes the path on the roadmap and handles the coordination inside the sector.

3.4.1 Topological layer

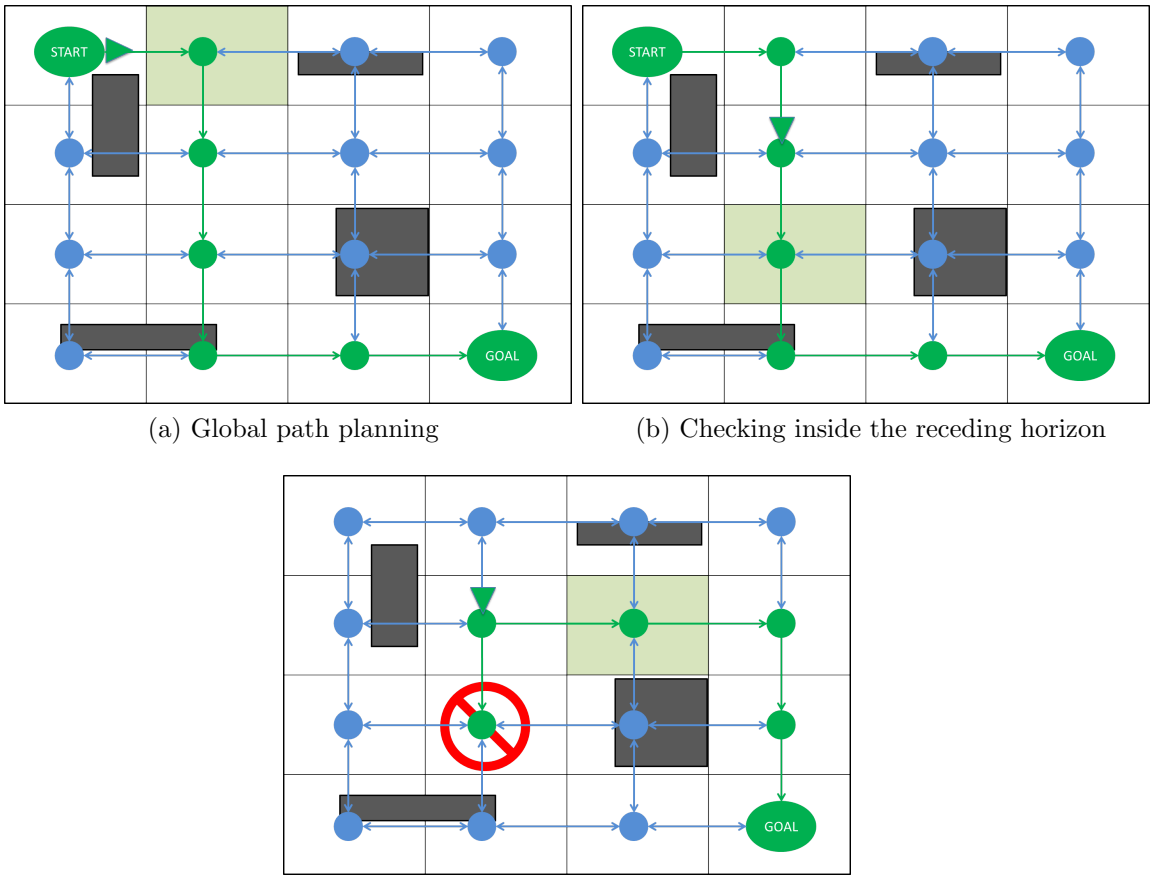
The first layer is the most abstract layer, and considers the topological representation of the roadmap as a set of sectors interconnected by a directed graph $\mathcal{Z}$ (see Section 3.3). A node of $\mathcal{Z}$ represents a sector, and an edge models the way a sector is connected to another defining its neighborhood.

The information owned by the sectors are used to plan the sub-optimal route for an AGV. Each vehicle has to reach its destination by minimizing a functional, such as such as the crossing time, average velocity, travel distance, etc. In this application, the AGVs have to complete their tasks in the shortest possible time, namely the total crossing time have to be minimized.

It is worth remarking that each AGV autonomously plans its path, as soon as it is assigned a destination. Hence, each AGV is in charge of computing its path, that is a sequence of sectors. In particular, this sequence connects the start sector with the goal one, exploiting the D* algorithm [48], combined with a MPC (Model Predictive Control [49]) mechanism. Namely, when entering a new sector, the AGV checks if the previously assigned path is still the optimal one. According to the MPC approach, at each step a prediction horizon is defined, along which the corresponding portion of the path is computed.

The graph $\mathcal{Z}$ is updated based on the status of the traffic, and then the generic weight $\omega_{i,j}$, defined as in Eq. (3.1), will change respectively. This combined approach is needed in order to re-plan the path in a dynamic manner. In particular the re-planning event is performed based on the traffic congestion (i.e. number of AGVs) of the planned sectors in the prediction horizon. Thus this process tries to avoid the congested sectors, and to minimize the effort for the local coordination requested subsequently within the sectors. It is worth noting that a congested area implies a higher number of negotiations and thus most of the time is spent waiting. This procedure is sketched in Fig. 3.5

This approach provides an optimal local solution but a sub-optimal global one, because only the part of the path inside the horizon is interested by the optimization. The procedure is summarized in Algorithm 3: in this algorithm, the vector path



(a) Global path planning

(b) Checking inside the receding horizon

(c) Re-planing of the path

Figure 3.5: The path planning on the topological layer: in 3.5a the path is searched by means the D* algorithm; in 3.5b the AGV moves along its path and checks the next sectors; in 3.5c a re-planning of the path is needed due to the new condition of the next sector; the graph $\mathcal{Z}$ is shown in blue.

contains the list of sectors in the planned path, the term H is the number of sectors that constitute the prediction horizon, and the index i identifies the current step.

3.4.2 Roadmap layer

Inside each sector the coordination among AGVs is needed. The second layer manages the real path following on the roadmap (created according to the procedure described in Section 3.3) and the avoidance of deadlocks and conflicts among AGVs or among AGVs and obstacles. The coordination is managed locally (in each sector) in a decentralized manner. With this hierarchical architecture it is possible to simplify the whole control in order to bound the coordination of the AGVs only inside each sector in a local way.

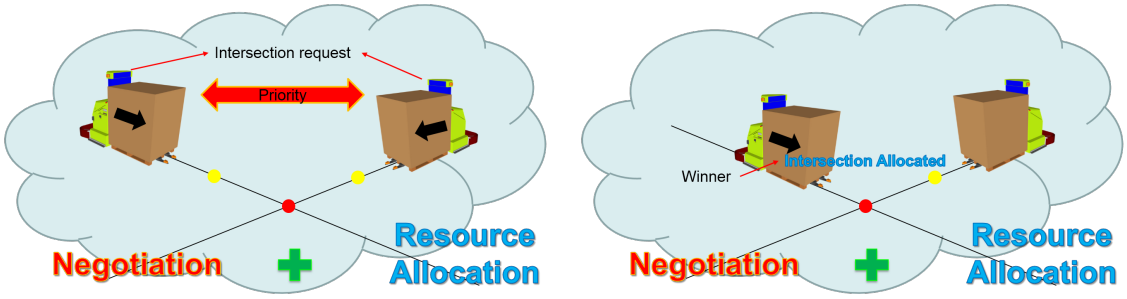
Each AGV has to compute a path to reach the next planned sector on the

Algorithm 3: Path planning on the topological layer

```
while path[i] ≠ goal do
    if Traffic Status is Changed then
        Update 𝒵;
        path[i : i + H] = D*(𝒵);
        go to path[i + 1];
    end
end
```

topological layer. The path planning inside a sector consists of assigning a set of segments to each AGV. The algorithm used to find the path is the simple A*. The choice is due to the fact that the roadmap is fixed, and local dynamic changes are not considered.

The path planning strategy is based on the assumption that each AGV has a prior knowledge of the geometry of the sectors and of the roadmap, according to Assumption A2. Conflicts among AGVs are managed by means of a hybrid approach, combining a *negotiation* mechanism with a *resource allocation* strategy, as pictorally shown in Fig.3.6. In particular, the resource (intersection area) is allocated only to a single AGV in order to avoid conflicts, and the negotiation permits to avoid deadlocks.



(a) Negotiation phase (b) Resource allocation

Figure 3.6: Roadmap Layer Coordination

This process is managed *locally*, because it takes place exclusively inside the sector: the AGVs share information among each other, without the participation of a centralized supervisor. The data exchange among AGVs concerns:

- *AGV priority*: each AGV is supposed to have a priority, related to the task it is carrying on. If this priority is not assigned a priori, several strategies can be found in the literature for decentralized priority allocation (see e.g. [50]).

Algorithm 4: Coordination on the roadmap layer

```
1  if pos(AGV[i]) = attention_point_j^q then
2  |   request[i, j] := true;
3  |   if ∃k ≠ i such that request[k, j] = true then
4  |   |   Negotiation;
5  |   else
6  |   |   AGV[i] := winner;
7  |   end
8  |   if AGV[i] = winner and status[A_q] = free then
9  |   |   move;
10 |   |   request[i, j] := false;
11 |   else
12 |   |   stop;
13 |   |   go to line 2;
14 |   end
15 end
```

- *Main segment request*: an AGV which is approaching an intersection has to communicate the intention of allocating its next segments to the others.
- *Intersection area request*: an AGV has to communicate the intention of allocating all the segment contained in intersection area.
- *Intersection area allocation*: an AGV that is allowed to go through an intersection area has to communicate this to the others.

The coordination procedure is described in details in Algorithms 4 and 5. In these algorithms, the term $\mathsf{pos}\,(AGV\,[i])$ represents the position of the i-th AGV, while $\mathsf{attention\_point}_{\mathsf{j}}^{\mathsf{q}}$ is the j-th attention point related to the q-th intersection area, identified as $\mathcal{A}_q$. The term $\mathsf{status}\,[\mathcal{A}_q]$ is the status of the intersection area $\mathcal{A}_q$. The term $\mathsf{request}\,[i, j]$ represents the request of the i-th AGV for the allocation of the q-th intersection area. The term $AGV\,[i]_p$ is the value of the priority of the i-th AGV and $\mathcal{A}_q^p$ is the value of the priority of the AGV that is winning the current negotiation for the intersection area q.

3.5 Experimental Validation

In order to evaluate the overall methodology, several simulations have been implemented in Matlab. Different real warehouse plants have been used to test the coordination algorithm and the ensemble approach.

Algorithm 5: Negotiation

```
1  if AGV[i]_p < A_q^p then
2  |   A_q^p := AGV[i]_p;
3  end
4  if AGV[i]_p > A_q^p then
5  |   return;
6  end
7  if AGV[i]_p = A_q^p then
8  |   AGV[i] := winner;
9  |   return;
10 end
```

First of all, the hierarchical planner has been compared to a simple A* planner on the same automatic roadmap. Each AGV computes its path with the A* algorithm by minimizing the Euclidean distance (the weight among sectors is not considered). The local coordination on the roadmap layer is still the same. The comparison has been carried out analyzing the total crossing time, and according to the following conditions:

- number of AGVs: 10, 20, 30. For each configuration, 10 runs of simulation have been performed
- maximum capacity of 4 AGVs allowed in each sector
- same tasks (randomly generated in each run) and initial positions for the compared methods
- the simulation stops when all the AGVs reach their goals and the queue of tasks is empty
- same priority scheme for the compared methods in each run

The results are collected in Fig. 3.7, where it is possible to note how the hierarchical planner performs better as the number of AGVs increases, due to less negotiations required in each sector. In fact, with the proposed hierarchical planner, the AGVs are lead up to change their paths in order to avoid congested sectors and thus to reduce the number of negotiations. For small numbers this behavior is not so pronounced because less changes of the original path are needed. It is worth noting that a negotiation represents a waiting time for the AGV which has to stop if it is the looser of negotiation, and thus less negotiations are preferable because

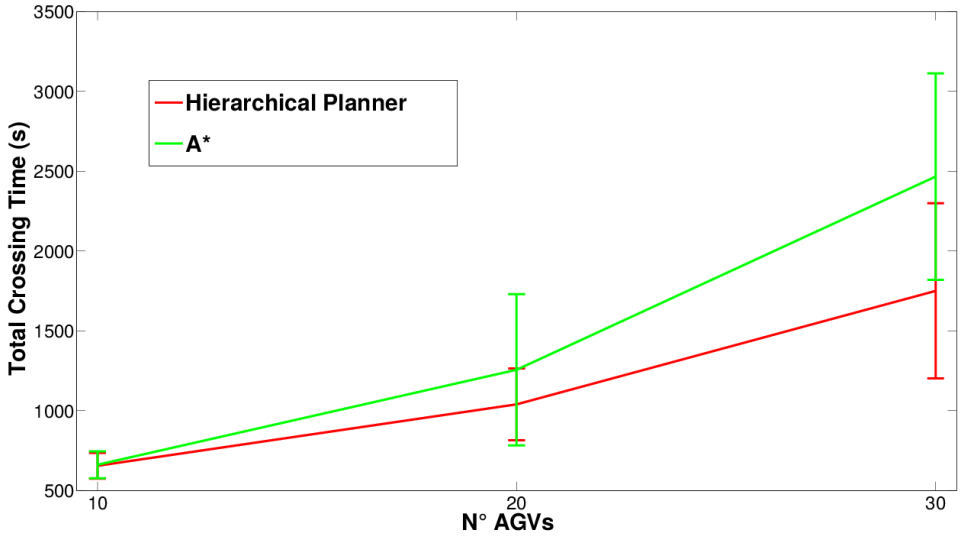


Figure 3.7: Evaluation of the proposed Hierarchical Planner

#AGV	h	p-value
10	0	0.8154
15	1	0.0140
20	1	0.0026

Table 3.1: Statistical validity of the comparison

they lead to a shorter total crossing time. Furthermore a p-values test has been conducted to prove the validity of the comparison where the null hypothesis that data from Hierarchical planner are a random sample from a normal distribution with mean came form the A* planner is tested against the alternative that the mean is not the previous one. The result of the test is shown in Table 3.1, where $h = 1$ indicates a rejection of the null hypothesis at the 5% significance level and $h = 0$ indicates a failure to reject the null hypothesis at the 5% significance level. The test confirms the validity of the previous comparison.

Fig. 3.8 shows the variance related to the same layout with an optimized roadmap, that is the connectivity is maximized, and with non-optimized roadmap. It is possible to state that the better performance are with the optimized roadmap and that the increasing of the variance is due to the high number of negotiations which, depending on the random priority of the AGVs, can provide different results on tests performed in similar conditions.

The *hierarchical planner* has been validated also by means of experiments in real industrial environment. The test has been conducted with 3 real AGVs navigating

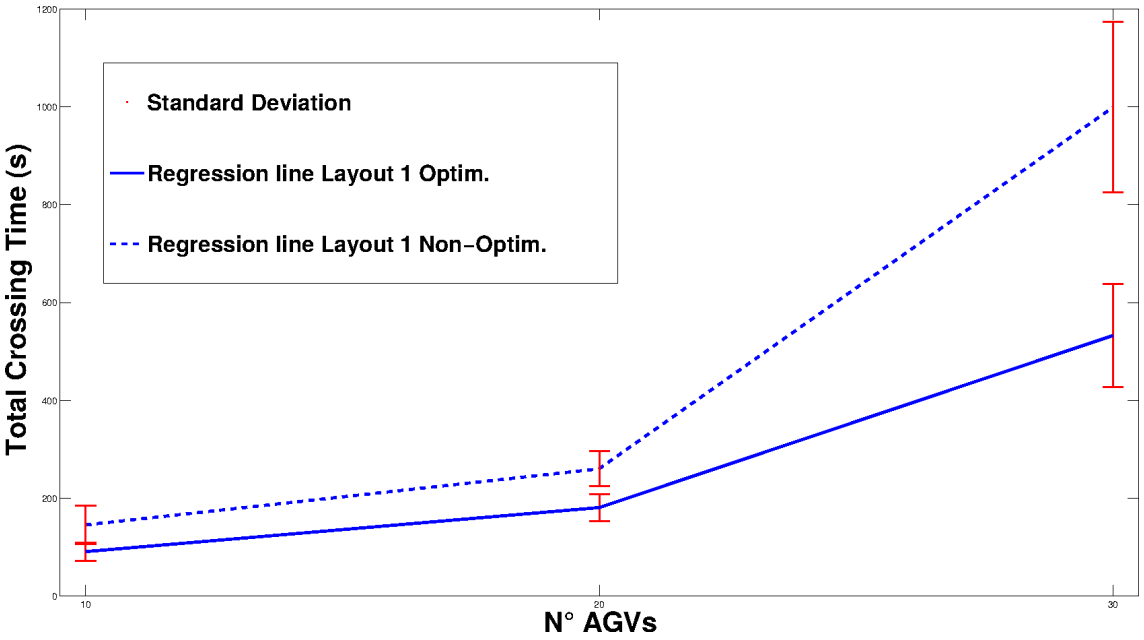


Figure 3.8: Total Crossing Time versus number of AGVs: variance

in a small warehouse. The sequence shown in Fig. 3.9 highlights the coordination on the roadmap layer.

3.6 Conclusion

This Chapter describes an ensemble strategy for the coordination of a fleet of AGVs through a two-layer control architecture. An ensemble approach is considered in order to improve the efficiency and the flexibility of the global system. On these lines, the automatic roadmap generation process proposed in Section 2 is merger with a coordination strategy. The roadmap algorithm is extended in order to match the automatic roadmap itself with the coordination mechanism, in particular the system is modeled as a lumped parameter model by the automatic roadmap algorithm.

The proposed coordination approach is based on a hierarchical architecture where a hierarchical path planning is achieved. The coordination is partially decentralized in the sense that local negotiation is performed exploiting inter-vehicle communication whereas the global path planning makes use of shared (i.e. centralized) information. Furthermore the simulations have shown that it is easily possible to manage a high number of vehicles. Since the proposed hierarchical planner performs better compared to a common planning algorithm on the same roadmap and the generated roadmap improves the traffic flow of the system, it is possible to state that the proposed method works well as an ensemble approach on the whole system.

(a) Initial configuration

(b) Negotiation phase

(c) The AGV on the left has to wait

(d) The AGV on the left moves as soon as the resource is free

Figure 3.9: Validation in a real warehouse

Chapter 4

Optimized Coordination Strategy

In Chapter 3 a hierarchical coordination strategy, called *hierarchical planner*, has been proposed. In particular, two layers are exploited. This Chapter focuses on the low level planner. In the previous Chapters, the coordination was based on a priority scheme applied to the fleet. It is worth noting that the choice of priorities tremendously affects the performance of the planner. In order to overcome to this issue, in this Chapter a new coordination strategy based on an optimization method for the low level planner is proposed.

4.1 Introduction

As explained in Chapter 3, the environment the robots operate in is partitioned into sectors. The sequence of sectors each AGV has to cross for reaching its destination is computed by a centralized planner to improve the performance. Within each sector, the AGVs are coordinated using a decentralized strategy which provides a simple and scalable traffic strategy. The centralized planner in each sector employs a Model Predictive Control (MPC) scheme. Each time an AGV enters a new sector, the central unit determines the traffic level in each sector by counting the number of AGVs it contains. Using the D^* [48] algorithm the sequence of sectors the AGV needs to track is recomputed based on the current traffic condition. The introduction of such a simple traffic measure has significantly increased the performance of the traffic manager, see [43] for more details and [51] for a industrial benchmark. The decentralized coordination within each sector is performed by means of a negotiation strategy among the AGVs based on a priority scheme. A shared resource, i.e., a road segment, is allocated to the winner of each negotiation round, while the loser of the negotiation round must wait until the shared resource becomes free. It is worth noting that this negotiation mechanism inevitably leads the AGVs to spend

most of their time waiting. Thus a drawback of these previous approaches is that vehicles required to cross a sector end up spending most of their time waiting for negotiations. Thus the amount of time spent locally, namely inside each sector, tremendously impacts the overall performance of the fleet.

This Chapter aims at overcoming this drawback by minimizing the number of interactions between AGVs. The idea is to minimize the amount of time vehicles spend negotiating complex traffic patterns within each sector as they navigate in the workspace while avoiding collisions with each another. This method considers the partitioning of an indoor environment into several sectors. Within each sector, a local coordination strategy, that relies on a centralized optimization approach, seeks to maximize the throughput of AGVs through the sector. The contribution of this Chapter is a complete formulation of the coordination problem as a convex optimization problem. In particular, the coordination problem is posed as a quadratic program (QP) where the crossing time for the vehicles, i.e., the time it takes for the vehicles to enter and leave a given sector, is minimized with respect to the vehicle velocities. Furthermore a complete analysis about the feasibility and optimality of the proposed method is introduced as well as an exhaustive experimental validation in real scenarios extracted from real automated warehouses with AGV systems. The proposed method extends the idea of [52] and the framework presented in [37, 43] is exploited. Some preliminary results in which a draft version of the method is applied to a virtual environment can be found in [52].

4.1.1 Outline

The Chapter is organized as follows: the problem formulation and key assumptions to the approach are stated in Section 4.2. Section 4.3 describes the optimization problem and the formulation of the problem as a quadratic program. Section 4.4 provides an analysis of the optimality and feasibility of the proposed strategy. Section 4.5 shows the implementation of the proposed method and Section 4.6 presents simulation and experimental results. Finally Section 4.7 summarizes key results.

4.2 Problem Statement

4.2.1 Preliminaries

In this section, we briefly summarize some key concepts from convex optimization which will be exploited in our analysis of the optimality and feasibility of the proposed strategy in Section 4.4.

Lemma 4.1. *[53, Section 2.5.1 - 5.9.4] For convex optimization problems, if $g : \mathcal{C} \to \mathbb{R}$ is a real-valued continuous strictly convex objective function on a non-empty compact (i.e. bounded and closed) feasible set $\mathcal{C}$, a solution always exists and it is unique.*

Lemma 4.2. *[53, Section 3.1.4] Given a twice differentiable function f, whose open domain is **dom** f, that is its Hessian $\nabla^2 f$ exists at each point in **dom** f, then f is convex if and only if **dom** f is convex and its Hessian is positive semi-definite:*

$$\nabla^2 f(x) \succeq 0 \quad \forall x \in \boldsymbol{dom} f.$$

Lemma 4.3. *[53, Section 4.2.2] For convex optimization problems any locally optimal point is also globally optimal. Namely, given a solution to the optimization problem, the solution is guaranteed to be optimal if the problem is convex.*

4.2.2 Scenario

This Chapter aims to coordinate a fleet of vehicles moving in industrial-like environments, such as automated warehouses. The motion of the fleet is constrained on a predefined roadmap $\mathcal{R}$, similarly to existing AGV systems. The roadmap is partitioned into a collection of segments

$$\mathcal{P} = \{p_1, \ldots, p_W\},$$

where W is the total number of segments in $\mathcal{R}$, and it can be naturally modeled as a strongly connected graph Υ, where the nodes are the segments and edges represent the links among the segments. A path π on $\mathcal{R}$ is simply a sequence of adjacent segments the AGV can track, or equivalently a sequence of nodes on the graph Υ. The terms *sequence of segments* and *sequence of nodes* are equivalent and will be used interchangeably in the rest of the chapter.

Since the environment and consequently the roadmap are partitioned into the set of sectors $S = \{S_1, \ldots, S_G\}$, where G denotes the total number of sectors, we define a sector as an area of the roadmap which can be distinguished from the other areas based on topological, logistic and geometric properties. An algorithm that automates the construction of a roadmap with the described features can be found in the recent work [42].

We define the finite number $\mathcal{Z}_q$ of paths that are completely enclosed in the q-th sector S_q as:

$$\mathbf{P}_q = \{\pi^1, \ldots, \pi^{\mathcal{Z}_q}\} \in S_q.$$

A path is a sequence of segments, thus the i-th path within S_q is composed of M_i segments, formally:

$$\pi^i = \{\pi_1^i, \cdots, \pi_{M_i}^i\}.$$

The set of all the segments enclosed within the sector S_q can be formulated as

$$\mathbf{S}_q = \{\pi_j^i,\ \ i = 1, \ldots, Z_q,\ \ j = 1, \ldots, M_i\}$$

and its cardinality is $\sharp\mathbf{S}_q = Z_q \times \sum_i M_i$. An example of the partitioning of a real workspace is shown in Fig. 3.2 and Fig. 2.2 pictorially depicts segments and paths within a sector.

To model the relative distance among neighboring vehicles, each vehicle is modeled as a 2D circle with a radius of $\frac{\delta}{2}$. Let $\mathcal{V}(\pi_h^i)$ denote the portion of space occupied by a vehicle moving along the segment π_h^i, that is the trace of the vehicle along that segment. A segment collides with another segment when their traces intersect, namely when the following condition holds:

Definition 4.1. ***Collision Condition*** *The segments π_h^i and π_r^j, of the i-th and j-th path respectively, are in collision if $\mathcal{V}(\pi_h^i) \cap \mathcal{V}(\pi_r^j) \neq 0$.*

Let Π be the operator which associates a segment to the path where the segment is contained, formally $\Pi(p) = i \mid p \in \pi^i$, where the notation p represents the generic segment in a sector. It is possible to introduce now the notion of *collision segments set* (CSS) of a segment as:

Definition 4.2. ***Collision Segments Set (CSS)*** $CSS(p) = \{\nu \in \mathbf{S}_q \mid \nu \neq p,\ \ \mathcal{V}(\nu) \cap \mathcal{V}(p) \neq 0\ \ \&\ \ \Pi(\nu) \neq \Pi(p)\}$.

Where $CSS(p)$ is the set relative to segment p.

Within a single sector, an intersection area $\mathcal{A}_q$ of S_q is the bounded region of a sector where the coordination problem occurs, and there exists exactly one intersection area for each sector if the roadmap is built according to [42]. An intersection area is then formally defined as:

Definition 4.3. ***Intersection Area*** $\mathcal{A}_q = \bigcup_{p \in \mathbf{S}_q} CSS(p)$

An intersection area is then the set of the colliding segments of different paths in the sector and each sector is chosen in such a way that it contains at least one intersection area.

As explained in [43], the sector partition provides a high-level coarse topological representation of the roadmap that allows the abstraction of some properties of the

fleet to make the planning problem more tractable. The traffic manager works on two layers. The first layer performs a high level finite-horizon centralized planner that computes the sequence of sectors each AGV has to cross for reaching its destination using a shortest path algorithm on the graph of sectors. In this chapter, the A^* algorithm is used to search for the path on the graph. The second layer represents a low level decentralized coordination strategy for the safe navigation inside each sector. Thus, coordination is only required within each sector, and in particular within an intersection area, and the number of interactions among the agents corresponds to the number of local negotiations among them. A shared resource, *i.e.*, a road segment, is then allocated to the winner of each negotiation round, while the losers of the negotiation round must wait until the shared resource becomes free. It is worth noting that the results of these negotiations are not known a priori and that they depend on the path and the priorities of each AGV. Thus, the system is not deterministic and the traffic control and management is not trivial. In particular, the negotiation process affects the total time a vehicle spends traversing a given sector. As such, planning minimum time paths between sectors is challenging since the actual delays generated by the negotiations are difficult to predict.

The proposed methodology seeks to overcome this challenge by avoiding the negotiations entirely. This is achieved by choosing the best speed for each AGV to traverse its path to minimize its total crossing time while avoiding conflicts with other vehicles, *i.e.*, collisions.

The problem is formally stated in Chapter 3, in particular by Problem 3.1). It is now reported.

Problem 4.1. Multi AGV Coordination *Given:*

- *a fleet of N AGVs,*
- *a roadmap $\mathcal{R}$ and a set of sectors S, and*
- *the initial and final configuration for all the AGVs,*

define a coordination strategy such that each AGV is able to move from its initial position to its assigned final position while

- *minimizing the total crossing time and*
- *avoiding conflicts with other AGVs.*

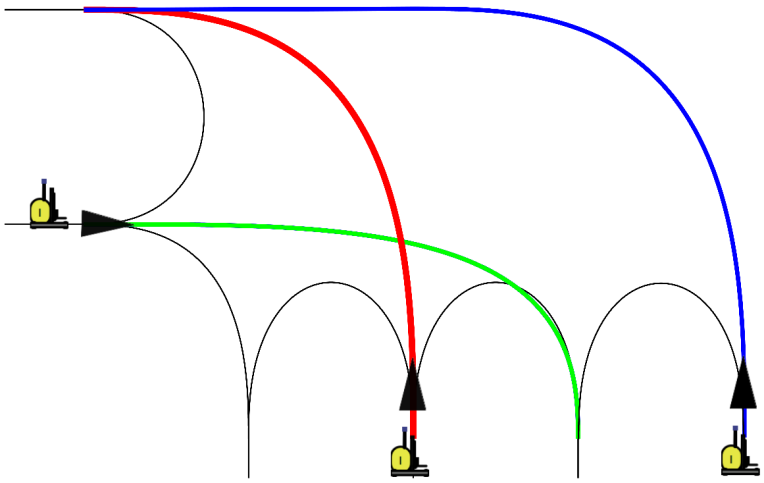


Figure 4.1: Multi AGV Coordination problem in an intersection area of a sector:

Fig. 4.1 shows a simple example of the problem where three AGVs have to follow the assigned paths such that conflicts, *i.e.*, collisions, are avoided. Each AGV starts in an initial position along its path and has to reach its own final position.

In the context of automated warehouses, the AGVs move in a controlled environment and the presence of unforeseen obstacles is not common. Thus it is not directly addressed in this work. Furthermore the AGVs are constrained to move on a roadmap and each segment of the roadmap has to be traversed with a constant velocity. We make the following **Assumptions**:

A1 No unforeseen events, such as the presence of dynamic obstacles (manual forklift, people, etc.) are considered.

A2 An arbitrary velocity along a path v_i is assigned to the AGV i such that $V^i_{min} < v_i < V^i_{max}$.

A3 The velocity along a segment is constant, as commonly used in current industrial scenarios.

A4 Each AGV has a different pair of initial and final positions since only one AGV is allowed to occupy a segment at a time.

Since the coordination is only required within each sector, hereafter a single sector scenario is used to describe the proposed methodology. In fact, it is worth noting that the coordination in each sector is independent with respect to the other

sectors. The overall performance of the proposed method is a linear combination of the performance of the coordination within each sector. In other words, the Problem 4.1 can be split among the sectors where it can be locally solved.

4.3 Modeling and Optimization Problem

This section aims at modeling the Multi AGV Coordination (Problem 4.1) as a QP problem. The output is a set of velocities that guarantee a safe minimum distance between all agents and that minimize the total time the AGVs need to accomplish the tasks. In general the problem can be solved considering each sector independently. In this section we show how the coordination problem within a single sector can be formulated as a convex optimization problem with feasible solutions.

4.3.1 Objective Function

Consider a sector S_q with N AGVs, the path of the i-th AGV is composed of $M_i \in S_q$ segments of length d_i^k, $k = 1, \dots, M_i$. Let us define

$$d_i = \sum_{k=1}^{M_i} d_i^k$$

as the length of the path of the i-th AGV. The velocity on the k-th segment for the i-th AGV is v_i^k. Then the *crossing time* for the i-th AGV, that is the time the AGV takes to exit the sector, is:

$$\Delta t_i = \sum_{k=1}^{M_i} \frac{d_i^k}{v_i^k}. \tag{4.1}$$

The highest crossing time among all the AGVs provides the *total crossing time* of the sector. It is formally given by:

$$\Delta T = \max_i \Delta t_i = \max_i \sum_{k=1}^{M_i} \frac{d_i^k}{v_i^k} \tag{4.2}$$

The quantity ΔT on a given sector has to be minimized. Let us define $\bar{M} = \sum_{i=1}^{N} M_i$, then

$$\mathbf{v} = [v_1^1, \cdots, v_i^{M_1}, \cdots, v_N^{M_N}]^T \in \mathbb{R}^{\bar{M}}$$

is a vector containing all the vehicle velocities. Equation (4.2) is non-linear with respect to the parameters $\mathbf{v}$. The minimization of ΔT maximizes the throughput of the sector, namely the smaller the *total crossing time* the more the number of AGVs that can cross the sector in a given amount of time. Since a path is already

assigned to each vehicle, the distance to travel is constant for each vehicle. As such, the only parameters to be chosen in (4.2) are the velocities v_i^k. It is important to note that minimizing the total crossing time is equivalent to maximizing the velocity on a fixed distance, or equivalently to minimizing its negative. Therefore, we can formulate the following objective function:

$$f(\mathbf{v}) = -\sum_{i=1}^{N}\sum_{k=1}^{M_i} v_i^k. \tag{4.3}$$

It is straightforward to show that (4.3) is both linear and convex with respect to the optimization variables $\mathbf{v}$.

4.3.2 Constraints

Two constraints are modeled: the first is on the velocity of each AGV, the second is on the safe distance between the vehicles. The velocities $\mathbf{v}$ are lower and upper bounded based on the properties of the vehicles (Assumption A2). The constraints are formalized as:

$$V_{min}^i < v_i < V_{max}^i, \;\; \forall i = 1, \ldots, \bar{M} \tag{4.4}$$

which are linear with respect to $\mathbf{v}$.

Furthermore, each AGV is modeled as 2D circle centered at the point $\mathcal{X}_i = [x_i, y_i]^T$ with a radius of $\frac{\delta}{2}$. Collision avoidance is guaranteed by ensuring that the relative distance, $\gamma_{i,j}$, between any two AGVs is grater than δ, namely:

$$\gamma_{i,j} = ||\mathcal{X}_i - \mathcal{X}_j|| = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} > \delta. \tag{4.5}$$

Eq. (4.5) describes the non-linear constraints for the pairwise distances between AGVs. For each pair of AGVs, (4.5) has to be computed for each pair of positions $(\mathcal{X}_i, \mathcal{X}_i)$ along their vehicle paths. This process is inefficient and increases the complexity of the optimization.

Consider an intersection area $\mathcal{A}_q$ and N vehicles moving on N different set of segments $\mathcal{K}_1^q, \ldots, \mathcal{K}_N^q$, that is $\mathcal{K}_i^q = \{\pi_h^i \in \mathcal{A}_q\}$ is the set of segments for the vehicle i. Then the sets of segments $\mathcal{K}_1^q, \ldots, \mathcal{K}_N^q$ are parameterized by $s_1 \in [0, d_1], \cdots, s_N \in [0, d_N]$, where s_i is the curvilinear abscissa used to denote an AGV's position along its trajectory, namely along the set of segments.

Let $\mathcal{S}^i(s_i) : \mathbb{R}^+ \to \mathbb{R}^2$ be a function which maps the curvilinear abscissa s_i to the Cartesian position $\mathcal{X}_i$ of the vehicle on its path. Let $\Delta X_{i,j}$ be the portion of the i-th path where vehicle i collides with vehicle j on the i-th path, *i.e.*, $\gamma_{i,j} < \delta$. Here

$\Delta X_{i,j}$ is a collision region for the i-th and j-th AGVs and can be formally defined as:

$$\Delta X_{i,j} = \left\{ s_i \mid \exists s_j \text{ such that } ||\mathcal{S}^i(s_i) - \mathcal{S}^j(s_j)|| \leq \delta \right\} \tag{4.6}$$

It is worth noting that the collision regions can be computed totally off-line since they are based solely on geometric information regarding the roadmap and the shape of the vehicles. The computational complexity of (4.6) for paths i and j is $O(M_i \times M_j)$ since each pair of segments $(\pi_h^i, \pi_r^j) \in \mathcal{K}_i^q \cup \mathcal{K}_j^q$ has to be checked.

Let $X_{i,j}^{min}$ and $X_{i,j}^{max}$ be the values of the curvilinear abscissa s_i corresponding to the start and final point respectively of the collision region $\Delta X_{i,j}$. Let us introduce $\omega_{i,j}^{min}$ and $\omega_{i,j}^{max}$ as the time that the vehicle i would take to reach the position $\mathcal{S}^i(X_{i,j}^{min})$ and $\mathcal{S}^i(X_{i,j}^{max})$ respectively. Furthermore the projection of $\Delta X_{i,j}$ on the time axis provides the set of collision time $\Omega_{i,j}$. Collision avoidance between the i-th and j-th AGVs is then satisfied when the following condition occurs:

$$\Omega_{i,j} \cap \Omega_{j,i} = \emptyset. \tag{4.7}$$

Fig. 4.2 pictorially depicts the described problem: with abuse of notation the vertical axis represents both the parameter s_i and s_j. The condition (4.7) implies that the set $\Omega_{i,j}, \Omega_{j,i}$, denoted by the red and blue line segments on the time axis in Fig. 4.2, are disjoint.

Equation (4.7) can be reformulated in terms of the midpoints and the total time defined by each set $\Omega_{*,*}$. In particular the distance between two mid-points has to be grater than the sum of the distances between the mid-point and one extreme point of both the sets. We introduce the following notation:

- $\alpha_{i,j} = \omega_{i,j}^{max} + \omega_{i,j}^{min}$
- $\beta_{i,j} = \omega_{i,j}^{max} - \omega_{i,j}^{min}$

The condition described by (4.7) can then be formalized as follows:

$$|\frac{\alpha_{i,j}}{2} - \frac{\alpha_{j,i}}{2}| > \frac{\beta_{i,j}}{2} + \frac{\beta_{j,i}}{2} \tag{4.8}$$

which simplifies to

$$|\alpha_{i,j} - \alpha_{j,i}| > \beta_{i,j} + \beta_{j,i} \tag{4.9}$$

Once (4.9) has been squared, the constraint becomes quadratic:

$$(\alpha_{i,j} - \alpha_{j,i})^2 > (\beta_{i,j} + \beta_{j,i})^2. \tag{4.10}$$

Let us now introduce the following *Lemma* which will show that (4.10) is quadratic with respect to $\mathbf{v}$. This will be instrumental to the formulation of the problem as a QP.

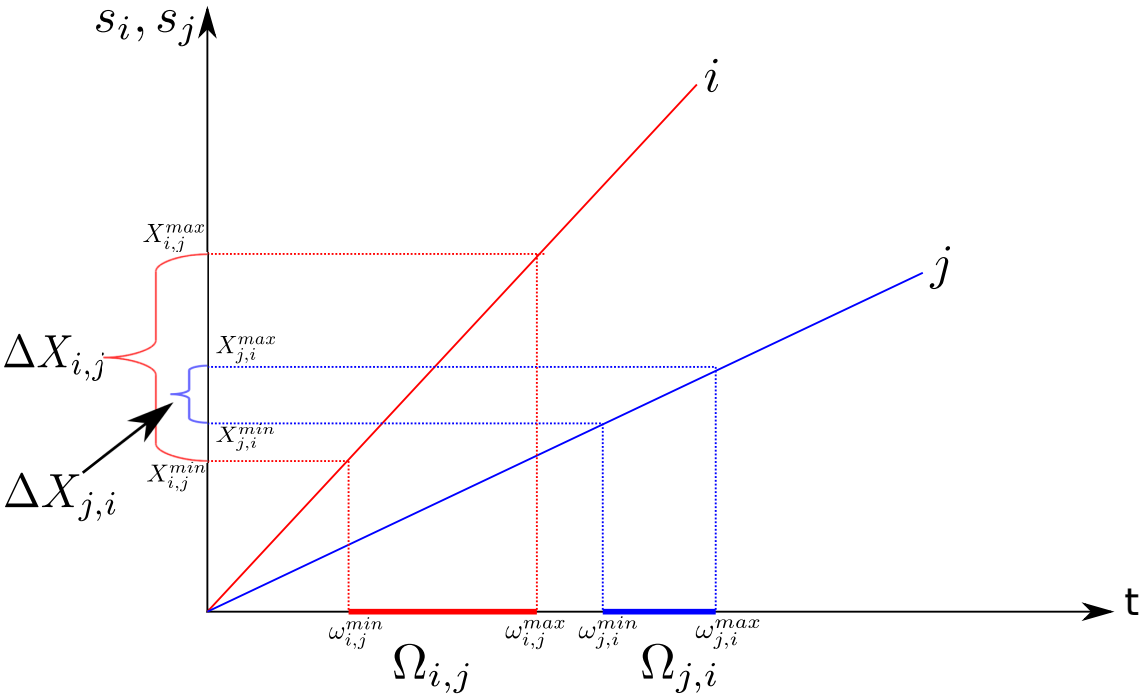


Figure 4.2: Coordination space in the case of two agents: the lines represent the velocities of the AGVs.

Lemma 4.4. *Eq.* (4.10) *is quadratic with respect to the velocities* v_i, v_j.

Proof. The first term of (4.10) can be reformulated as:

$$(\alpha_{i,j} - \alpha_{j,i})^2 = (\frac{X_{i,j}^{max} + X_{i,j}^{min}}{v_i} - \frac{X_{j,i}^{max} + X_{j,i}^{min}}{v_j})^2$$

$$(\alpha_{i,j} - \alpha_{j,i})^2 = (\frac{a_{ij}}{v_i^2} + \frac{a_{ji}}{v_j^2} - \frac{b_{ij}}{v_i v_j}).$$

Where a_{ij} and b_{ij} are opportune scalar constants, defined formally as:

$$a_{ij} = (X_{i,j}^{max} + X_{i,j}^{min})^2 \tag{4.11a}$$

$$b_{ij} = 2(X_{i,j}^{max} + X_{i,j}^{min})(X_{j,i}^{max} + X_{j,i}^{min}). \tag{4.11b}$$

In the same way, the second term can be formulated as:

$$(\beta_{i,j} + \beta_{j,i})^2 = (\frac{c_{ij}}{v_i^2} + \frac{c_{ji}}{v_j^2} + \frac{d_{ij}}{v_i v_j}).$$

Where c_{ij} and d_{ij} are opportune scalar constants defined as:

$$c_{ij} = (X_{i,j}^{max} - X_{i,j}^{min})^2 \tag{4.12a}$$

$$d_{ij} = 2(X_{i,j}^{max} - X_{i,j}^{min})(X_{j,i}^{max} - X_{j,i}^{min}). \tag{4.12b}$$

Considering $v_i, v_j > 0$ it is possible to multiply each term by $(v_i v_j)^2$ and the following quadratic expression is obtained:

$$v_j^2 a_{ij} + v_i^2 a_{ji} - v_i v_j b_{ij} > v_j^2 c_{ij} + v_i^2 c_{ji} + v_i v_j d_{ij} \tag{4.13}$$

□

4.3.3 Quadratic Constraints Linear Programming

The coordination within a single sector is modeled as an optimization problem using a linear objective function, and a set of quadratic constraints with linear constraints on the boundary. The optimization problem is given by:

$$\text{minimize} \quad -\sum_{i=1}^{\mathcal{N}} \sum_{k=1}^{M_i} v_i^k \tag{4.14a}$$

$$\text{subject to} \quad V_{min}^i \leq v_i \leq V_{max}^i, \ \ \forall v_i \in \mathbf{v} \tag{4.14b}$$

$$\text{and} \quad v_i^2 e_{ji} + v_j^2 e_{ij} + v_i v_j f_{ij} \leq 0, \ \ \forall i, j, \ \ i \neq j \tag{4.14c}$$

where (4.14c) is quadratic according with Lemma 4.4 and it is obtained from (4.13) with $e_{ij} = c_{ij} - a_{ij}$ and $f_{ij} = d_{ij} + b_{ij}$ as scalar constant values. Then the expression can be more compactly written in the following standard form [53].

$$\text{minimize} \quad f(\mathbf{v}) \tag{4.15a}$$

$$\text{subject to} \quad A\mathbf{v} \leq b \tag{4.15b}$$

$$\text{and} \quad 1/2\mathbf{v}^T H^{ij} \mathbf{v} \leq 0 \ \ \forall i, j = 1, \ldots, \bar{M}, \ \ \text{with} \ \ j > i \tag{4.15c}$$

where $A \in \mathbb{R}^{2\bar{M} \times \bar{M}}$ and $b \in \mathbb{R}^{2\bar{M}}$ encode velocity bounds in (4.14b). The matrix $H^{ij} \in \mathbb{R}^{\bar{M} \times \bar{M}}$ represents the inequality constraint in (4.14c) for the pair (i, j) and it is defined such that the element (l, p) is:

$$H_{l,p}^{ij} = \begin{cases} -2e_{ji} & \text{if } l = p = i \\ -2e_{ij} & \text{if } l = p = j \\ f_{ij} & \text{if } (l = i \ \text{ and } \ p = j) \ \text{ or } \ (p = i \ \text{ and } \ l = j) \\ 0 & \text{otherwise} \end{cases} \tag{4.16}$$

4.4 Analysis

This section demonstrates the optimality and the feasibility of the proposed coordination problem. We exploit the key concepts from *Convex Optimization* presented in Section 4.2.1.

Proposition 4.1. *The optimization problem given by* (4.15) *always has a solution.*

Proof. Let us define $\mathcal{C}$ as the feasible set of the optimization problem, namely

$$\mathcal{C} = \{\mathbf{v} \mid (A\mathbf{v} - b \leq 0), (1/2\mathbf{v}^T H^{ij}\mathbf{v} \leq 0 \ \ \forall i, j)\}$$

C is then a compact and convex set, since it is a union of non-decreasing, convex and compact sets. Furthermore, it is always possible to choose appropriate values of V^i_{min} and V^i_{max} for each vehicle i so that the set C is non-empty. For instance, in the extreme condition when $V^1_{min} = \cdots = V^N_{min} = 0$ and $V^1_{max} = \cdots = V^N_{max} = Inf$ the set C is certainly non-empty. Thus the optimization problem holds exactly one solution according to Lemma 4.1. □

Proposition 4.2. *The solution of the problem given by* (4.15) *is optimal.*

Proof. Let us consider first the objective function $f(\mathbf{v})$ in (4.15a). The domain of the function is $\mathbf{dom}\ f = \{v \in \mathbb{R}\}$, and thus it represents a convex set. Since

$$\mathbf{v} = -\sum_{i=1}^{\mathcal{N}} \sum_{k=1}^{M_i} v_i^k,$$

it is possible to note that the objective function is a collection of sums of linear and convex functions, then it is convex. Let us now consider the linear constraints given by (4.15b). It is possible to note that the constraints are affine and thus convex, essentially they define a bounded convex set in the solution space. The convexity of the constraint given by (4.15c) can be verified by means of its Hessian, defined as:

$$\nabla^2(1/2\mathbf{v}^T H^{ij}\mathbf{v}) = H^{ij}.$$

It is worth noting that the rank of H^{ij} is $rank(H^{ij}) = 2$ for each pair (i, j). Furthermore, since e_{ji} and e_{ij} are non-positive values and f_{ij} is always positive, the eigenvalues of H^{ij} are always non-negative or:

$$eig(H^{ij}) \geq 0.$$

Then the matrix H^{ij} is a positive semi-definite matrix, namely $H^{ij} \succeq 0$ for each pair (i, j). It now possible to state that the function (4.15c) is convex according to Lemma 4.2. Furthermore the whole optimization problem (4.15) is convex and thus, according to Lemma 4.3, any solution of the problem is also the optimal one.

□

It is worth remarking that the proposed method always gives a solution and this solution is also optimal. Mathematically the feasibility and the optimality of the proposed approach are thus proved. Furthermore, these results are further supported by several empirical results presented in the following sections which confirm the validity of the proposed method under most realistic conditions.

4.5 Implementation

The proposed method aims at coordinating a fleet of agents on a map composed by several sectors. For this reason, each sector is provided with a dedicated process which manages the optimization routine in order to implement the proposed methodology. The optimization algorithm runs independently in each sector whenever a new agent enters or leaves the intersection area.

Some variations are introduced in order to implement the methodology among several sectors and thus among several intersections. The methodology proposed in Section 4.2.2 is based on the assumption that all the vehicles are at the initial position of their paths in the sector. The optimization algorithm runs whenever a new agent enters or leaves the intersection area. Thus, in general, at each computation of the algorithm a vehicle i can be at any position along its path, not necessary the initial position (where $s_i = 0$).

Algorithm 6 shows the actual implemented method where $\mathcal{Q}_{new}(\mathcal{A}_q)$ represents the current list of vehicles contained the intersection area $\mathcal{A}_q$, and $\mathcal{Q}_{old}(\mathcal{A}_q)$ represents list of vehicles at the previous iteration of the algorithm. The starting position of each vehicle along its path, namely the current position at the instant of iteration of the algorithm, is collected in the vector $init(\mathcal{A}_q)$. Finally $\mathbf{v} \in \mathbb{R}^N$ is the vector of the computed N optimized velocities provided by solving the optimization problem. Algorithm 6 is performed at each sample time τ.

Algorithm 7 describes the single implemented controller for the i-th vehicle. It is worth noting that each AGV is controlled in a decentralized manner. Thus, each AGV entering a new intersection area $\mathcal{A}_q$ updates its position on the vector $init(\mathcal{A}_q)$ in order to get the correct and optimized velocity $\mathbf{v}(i)$. The i-th AGV stays in the queue $\mathcal{Q}_{old}(\mathcal{A}_q)$ and $\mathcal{Q}_{new}(\mathcal{A}_q)$ until it exists the intersection area.

4.6 Validation

The proposed optimized coordination is validated by means of comparison with the coordination strategy first presented in [37] and reported in Chapter 3 which relies

Algorithm 6: Optimized Coordination Main Routine

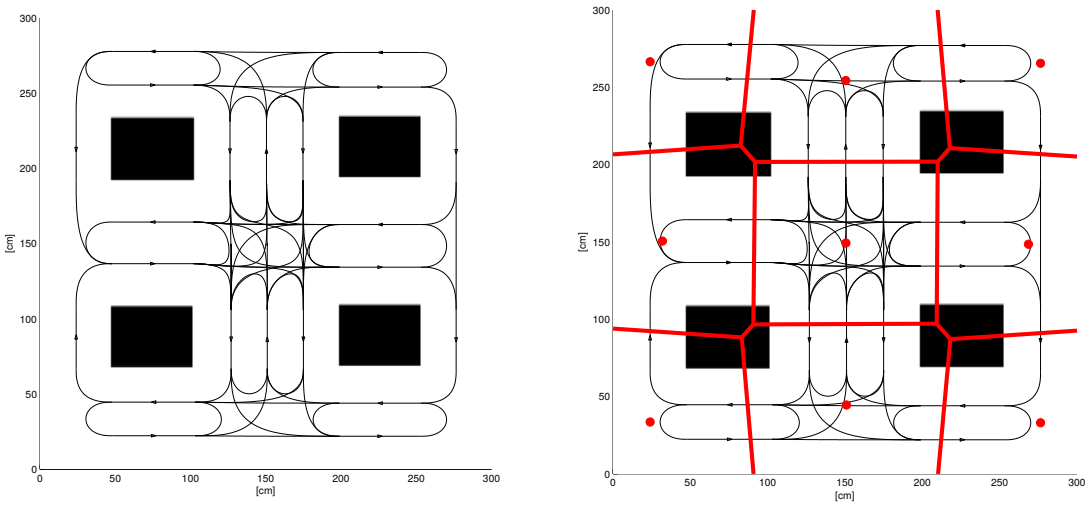
```
foreach A_q do
    if Q_new(A_q) ≠ Q_old(A_q) then
        Q_old(A_q) = Q_new(A_q) ;
        get init(A_q) ∀ paths π ∈ A_q ;
        v ← solve Eq. (4.14) ;
    end
    else
        continue;
    end
end
```

Algorithm 7: Agent's controller

```
Data: AGV i
while do
    if i enters A_q then
        Q_new(A_q) ← i ;
        get v(i) ;
    end
    else if i is moving within A_q then
        update init(A_q) of i along path π_i ;
        get v(i) ;
    end
    else
        remove i from Q_old(A_q) ;
        remove i from Q_new(A_q) ;
    end
end
```

on local negotiations. Hereafter we refer to the current proposed methodology as the *Optimized Strategy*, and to the other one as *Negotiated Strategy*. A decoupled optimal priority scheme [54] is applied to the *Negotiated Strategy* in order to obtain a better comparison. In other words, the priorities are assigned in an optimal manner in order to minimize the total crossing time.

The simulations are performed on different single intersections extracted form the roadmap of real warehouses, as shown in Fig. 4.4. Subsequently experimental tests are performed on a complete real environment. Fig. 4.3a shows a simple real environment composed by four rectangular obstacles and nine sectors. A roadmap is built for that environment using the algorithm described in Chapter 2.



(a) Roadmap (b) Sectors

Figure 4.3: Real map used in the simulations

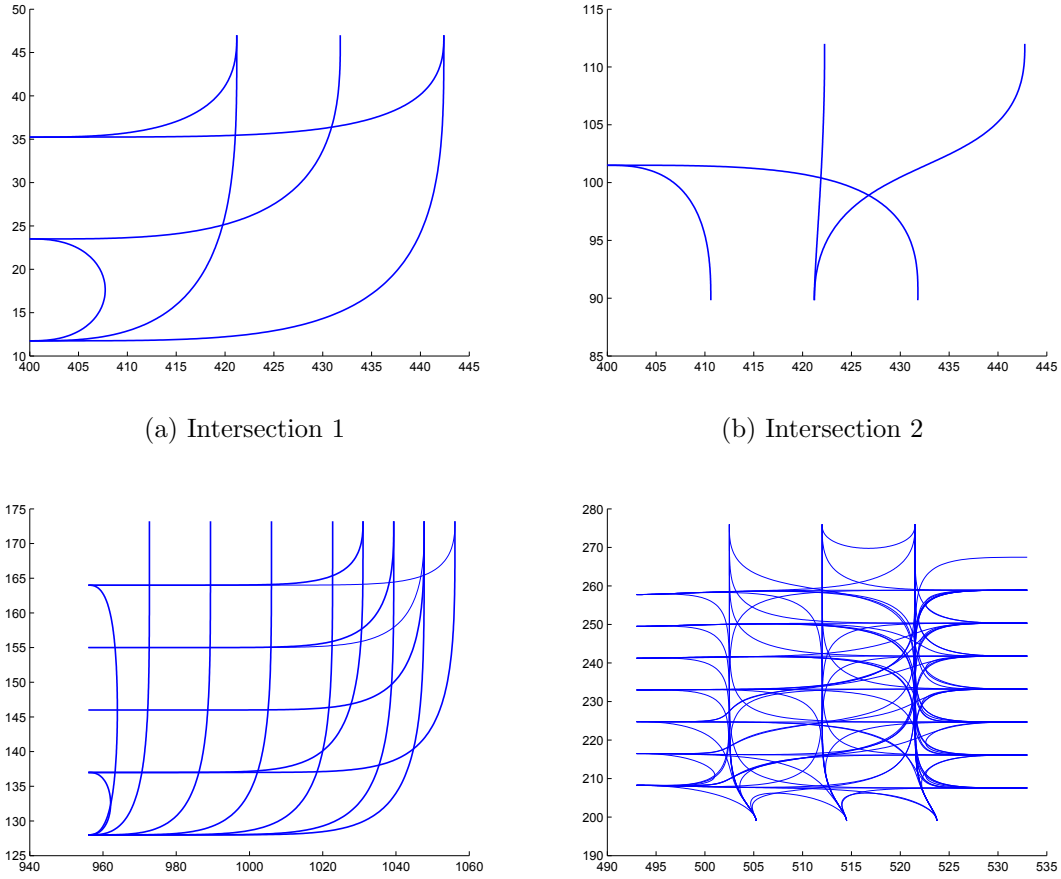
4.6.1 Simulations

The topological complexity of the roadmaps is different in each scenario which depends on the number of possible paths, the number of possible interactions, and physical size of the environment. Repeated tests have been conducted under the following conditions:

- 4 topology of intersection
- number of AGVs $\in [2, 11]$,
- the simulation stops when the intersection area is cleared,
- the priorities [37] for the *Negotiated Strategy* are optimized each time,
- the same paths assigned to the AGVs are considered for the comparison, and
- the paths are assigned randomly.

Ten simulation runs were performed for each configuration.

To compare, we consider the time needed for all AGVs to clear the sector or the maximum of the crossing time ($t_{clearing}$). We also considered the worst waiting time (t_{wait}), the average waiting time ($\bar{t}_{wait}$), and the computational time needed to obtain a solution (t_{calc}). The waiting time is the time that the AGVs have to wait in the same position, *i.e.*, $v_i = 0$, when yielding to other robots with higher priorities at an intersection in the roadmap. The worst waiting time is the maximum waiting



(a) Intersection 1

(b) Intersection 2

(c) Intersection 3

(d) Intersection 4

Figure 4.4: Different intersections used during the simulations

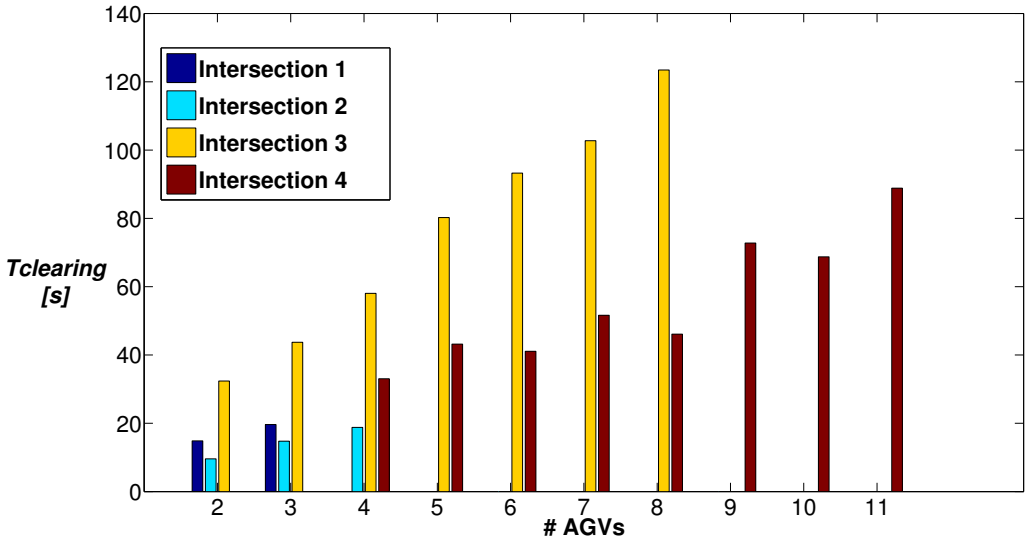
time for all the AGVs in the sector. The computational time is the actual time to compute a solution to the centralized optimization problem. The worst waiting time and the average waiting time are computed for the *Negotiated Strategy*, while the computational time is computed for the *Optimized Strategy*.

The simulations were performed in Matlab with the standard optimization solver. The results are summarized in Fig. 4.5.

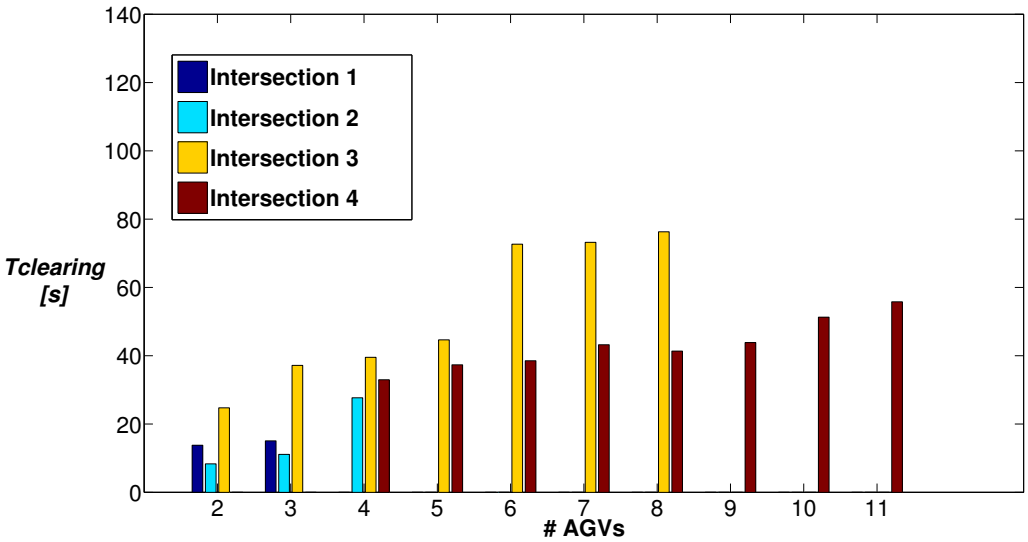
4.6.2 Experiments

The proposed approach has been implemented on a real indoor environment. The experimental environment and the implemented roadmap are shown in Fig. 4.6. The roadmap is divided into several sectors as shown in Fig. 4.3a.

The robots are differential-drive robot equipped with an on-board processor and equipped with odometry, an RGB-D sensor, and wifi networking capabilities. Local-



(a) *Negotiated Strategy*



(b) *Optimized Strategy*

Figure 4.5: Average of the Clearing Time versus number of AGVs in the different single intersections on 10 runs.

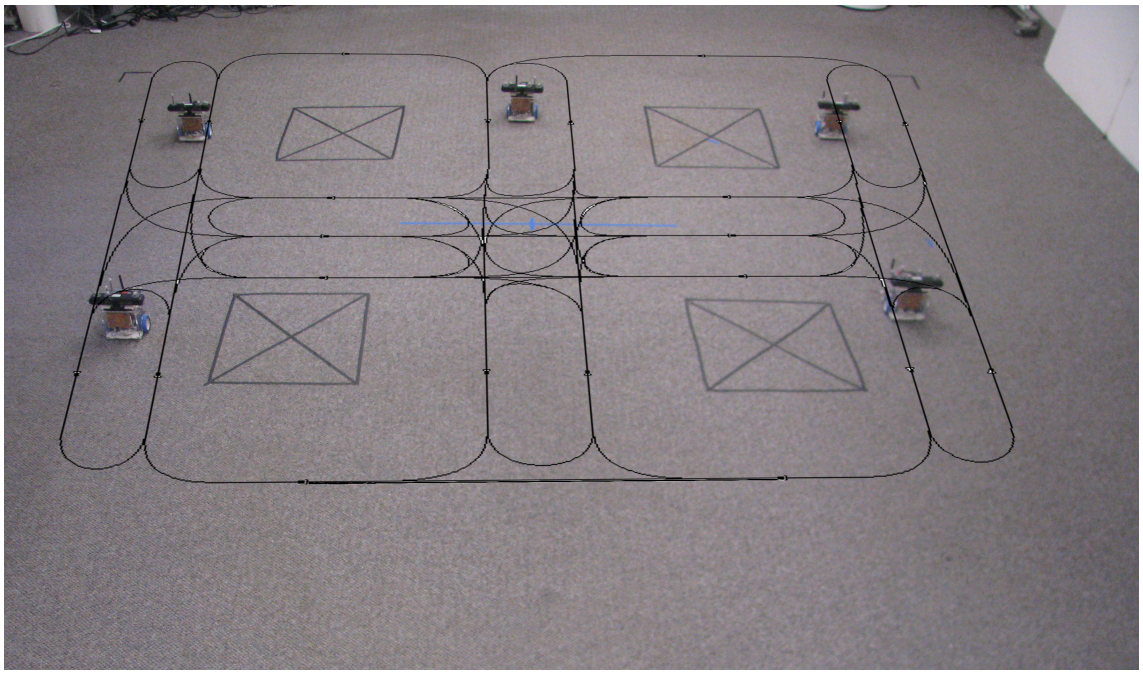


Figure 4.6: Real environment used for the experimental validation.

ization for the vehicles is provided by an external motion capture system. Fig. 4.7 shows the implementation architecture of the system. The core of the system, namely the *Optimized Coordination*, is implemented in the Matlab workspace. Then each robot is controlled by means of an independent ROS module which manages the tracking of the trajectories of the roadmap.

The experiments aim at validating the proposed optimized coordination method in a real scenario. Some simulated robots are used together with the real ones in order to increase the number of the fleet. The validation has been conducted under the following experimental set-up:

- 5 real robots
- 3 to 15 simulated robots
- tasks randomly assigned both to real and simulated robots

The main difference compared to the single intersection simulation is that the coordination of the fleet is based on the hierarchical control architecture explained in [37]. The proposed method (*Optimized Strategy*) is still compared to the *Negotiated Strategy* with respect to the time needed for all AGVs to reach their final destination, that is t_{total}. Fig. 4.8 shows the trend of the t_{total} with respect the number of AGVs on both of the strategies.

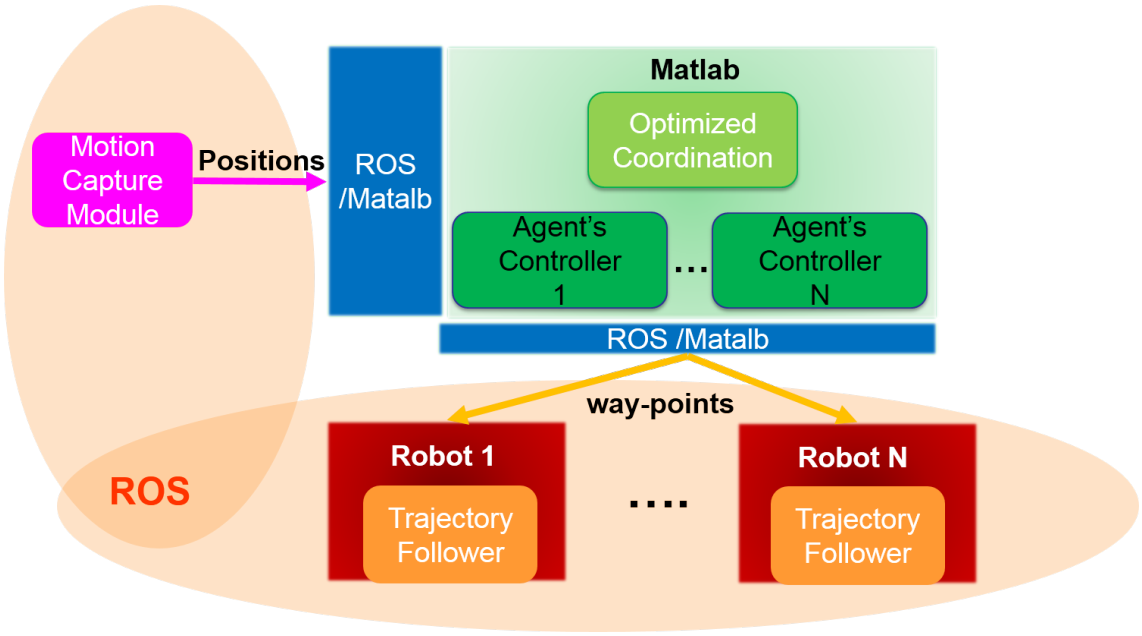


Figure 4.7: Implementation and architecture of the system

The data was also evaluated in terms of significance. Table 4.1 shows the statistical evaluation where $h = 1$ indicates a rejection of the null hypothesis at the 5% significance level and $h = 0$ indicates a failure to reject the null hypothesis at the 5% significance level.

#AGV	h	p-value
3	1	0.0346
5	1	0.0422
10	1	0.0163
15	1	0.0249

Table 4.1: Statistical validity of the comparison

4.7 Discussion and Conclusion

The simulations about the single intersection scenario show that the average clearing time (or the maximum crossing time) is always smaller using the *Optimized Strategy* rather than the *Negotiated Strategy* and clearly shows that the proposed methodology performs better than the previous one. Fig. 4.9 shows the relationship between the waiting time and the time a vehicle is actually moving, *i.e.*, $v_i > 0$, in the sector for the *Negotiated Strategy*. Overall, the time a vehicle spends waiting or idling is considerable when the *Negotiated Strategy* is used. In particular, the results show

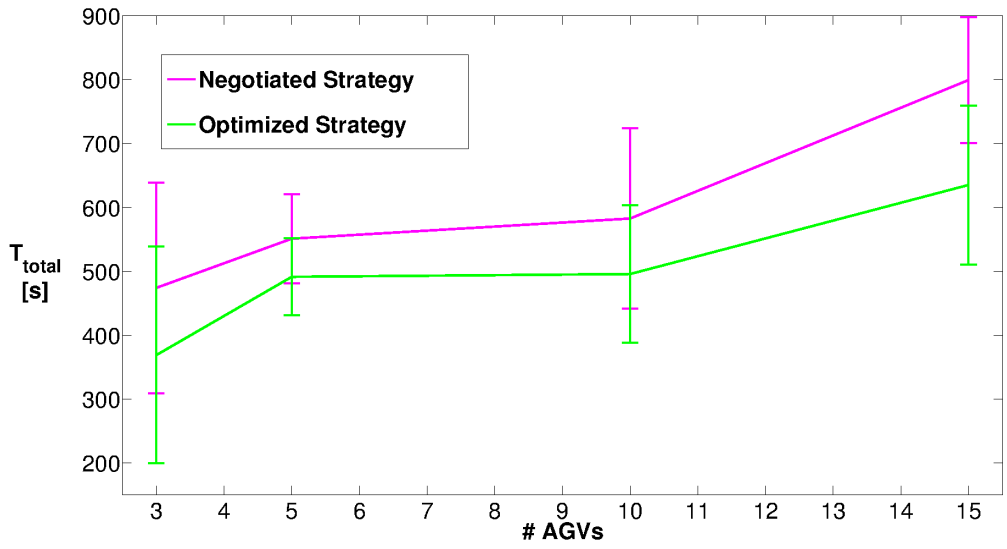


Figure 4.8: Total time versus number of agents in the complete map with standard deviation

that the average waiting time is almost 50% of the total time a vehicle spends within the sector and the worst waiting time is up to 90% of the total time. One can conclude that under the *Negotiated Strategy* the time an AGV spends at an intersection is mostly spent on waiting for its turn to cross. It is important to note that by construction the waiting time for each vehicle under the *Optimized Strategy* is always null. Here, coordination is achieved by managing the relative velocities of the agents to ensure collision avoidance while constraining the velocities to be always higher than zero.

When considering the average per vehicle waiting time resulting from the *Negotiated Strategy*, the computational time of the *Optimized Strategy* is essentially insignificant (see Fig. 4.10). The optimization algorithm does not suffer from a high computational burden despite being implemented in a centralized manner. This claim is supported by the fact that the strategy is designed to work with specific roadmaps and thus always tuned to the worst case scenario (Fig. 2.10). Furthermore, the required computation burden still stays low when the number of segments increases. The Optimized Strategy is concerned only with the number of vehicles since the collision regions are computed totally offline. Fig. 4.11 shows a comparison between the scalability of the *Negotiated Strategy*, which is almost linear, and the *Optimized Strategy*. We note that in the *Optimized Strategy* the maximum crossing time does not linearly increase with the number of agents but rather its trend is piecewise linear. This suggests that the *Optimized Strategy* can potentially be fur-

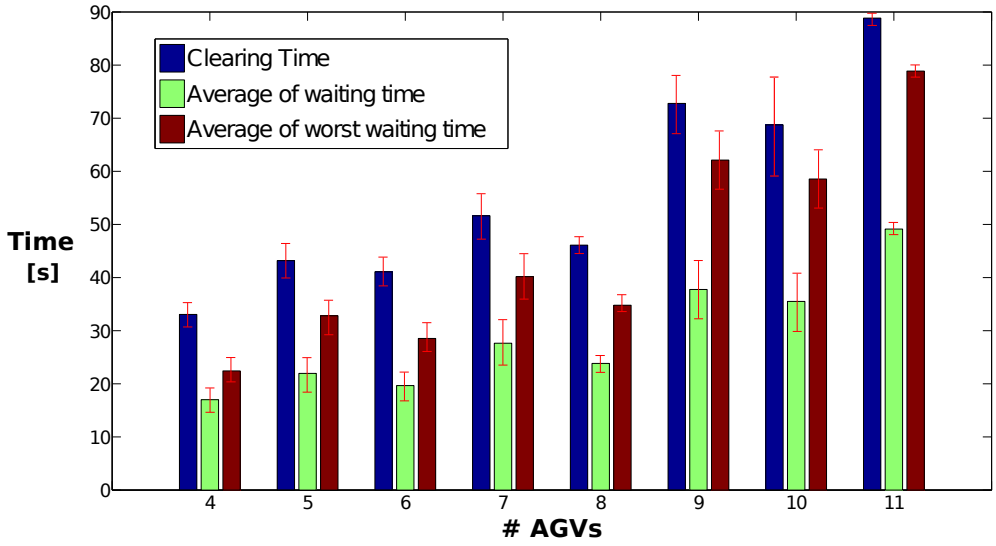


Figure 4.9: Waiting time versus clearing time for the *Negotiated Strategy*. Data presented for Intersection 4.

ther exploited when the system is complex. This is further supported by the fact that the waiting time for the *Negotiated Strategy* is two order of magnitude higher than the computational time for the *Optimized Strategy*.

For the complete scenario and for the real experiments, the results are very similar to the previous simulation. In particular t_{total} in the *Optimized Strategy* is always less then in the *Negotiated Strategy*. In addition, on a complete and real map the *Optimized Strategy* works better in term of total time. It is worth noting that the coordination and the performances within a sector are, in general, independent of the other sectors. Thus the complete behavior of the proposed methodology on the full map is approximately a linear combination of the local behavior within each sector.

Lastly, Fig. 4.8 shows the difference between the total time between the two strategies increases as the number of AGVs gets bigger. As the number of vehicles increases in the map, the higher the difference between the total time of the two strategies. While the complexity of the scenario affects the total time, the *Optimized Strategy* consistently performs better in the presence of more agents. The experiments also show that the proposed method works well under realistic conditions, where noise and disturbances normally arise (e.g. localization, delay, etc.).

To conclude, this Chapter has shown an optimized coordination strategy which aims at minimizing the time a fleet of AGVs takes to traverse different sectors. The problem is faced by solving a quadratic optimization process for each sector

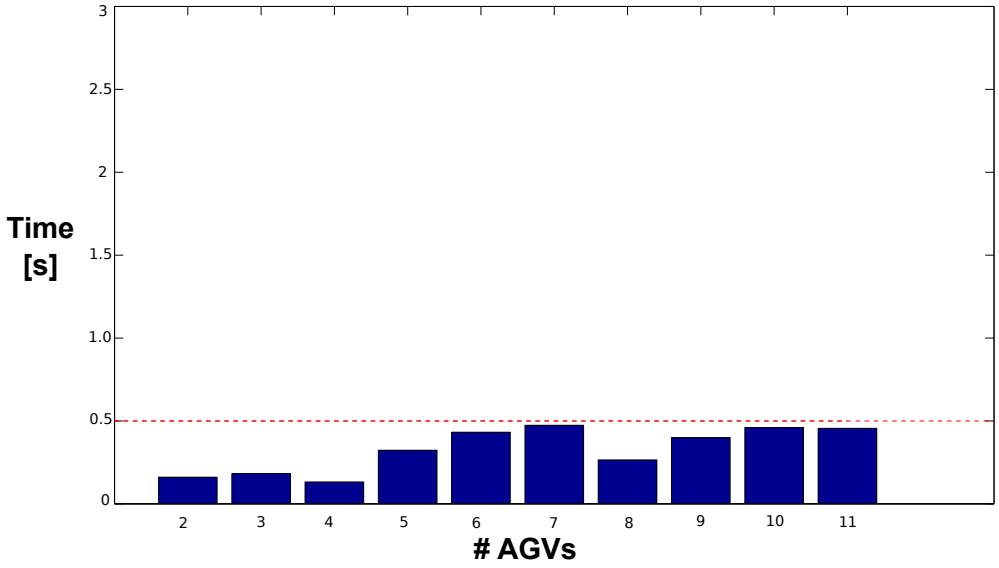


Figure 4.10: Computational time for the optimized strategy with respect to the number of AGVs.

of the roadmap and the quadratic problem can be solved in a very efficient way by standard solver programs. The optimal set of velocities is then applied to the AGVs moving in each sector in order to maximize its throughput. The simulations and experiments show that the proposed optimized method significantly outperforms the decentralized negotiated strategy.

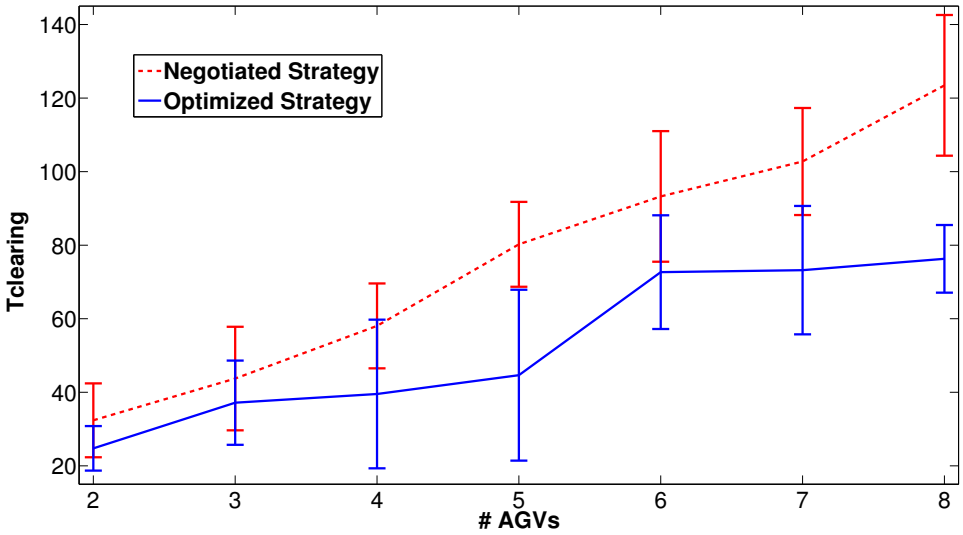


Figure 4.11: Clearing Time versus number of agents in the Intersection 3 with standard deviation

Chapter 5

Dynamic Mission Assignment

In this Chapter a methodology for mission assignment in multiple vehicle system exploited in industrial environments for logistics operations is introduced. The model of the traffic is explicitly considered for performing an optimized dynamic mission assignment. Simulation results are provided for comparing the proposed methodology with state-of-the-art techniques, that do not consider the state of the traffic while allocating missions.

5.1 Introduction

Each transportation task is defined as a *mission*. The *mission assignment* problem consists then in assigning a mission to be completed to each AGV. Clearly, in order to optimize the overall performance of the system, it is necessary perform the assignment in an optimized manner, that means minimizing the overall completion time. Mainly utilized assignment methods consider the mission assignment problem in a *static manner*: roughly speaking, each mission is assigned to the closest AGV. This assignment method does not consider the dynamic effects due to the motion of the AGVs themselves: namely, traffic jams might decelerate the AGVs, thus making the assignment very far from the optimal solution. In this Chapter a traffic model is used within the assignment problem: specifically, information on the state of the system is utilized to obtain a quantitative measure of the traffic, that is then exploited for an optimized mission assignment.

In this Chapter we consider the mission assignment problem for multi-AGV systems used for factory logistics: the architecture of the system under consideration is depicted in Fig. 5.1. In particular, we consider the case where AGVs are exploited for goods transportation inside a manufacturing plant. In this case, each *mission* to be accomplished consists in taking a particular batch of goods (e.g. a box or a

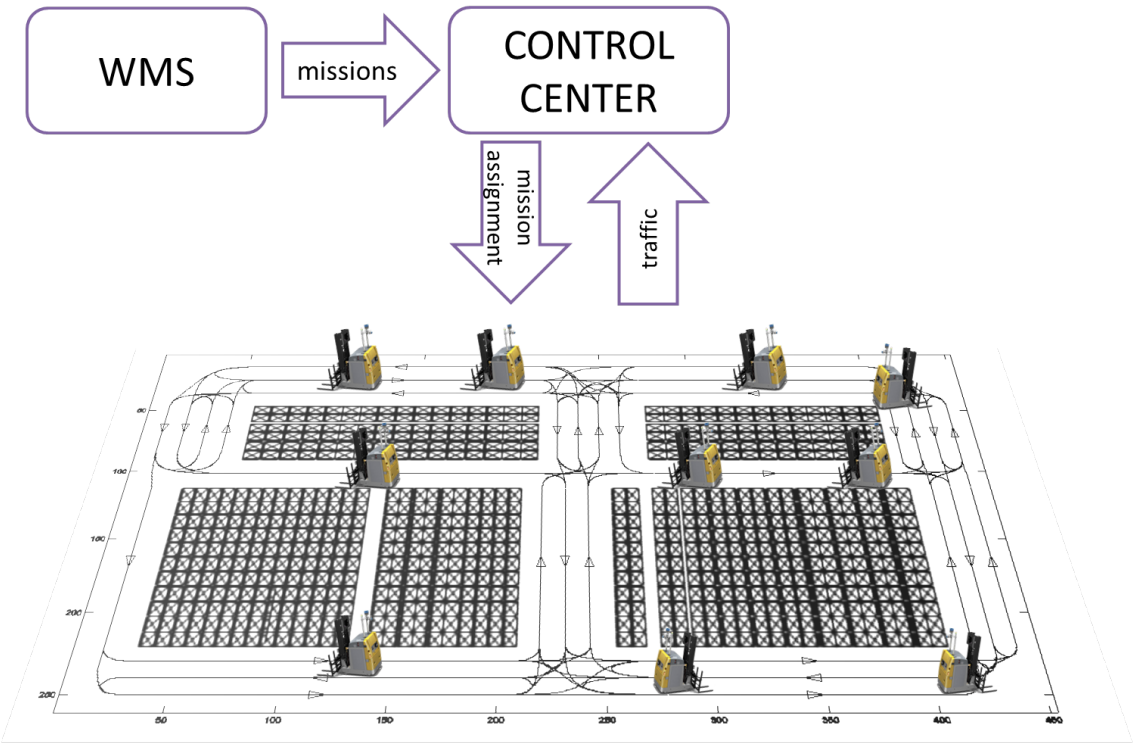


Figure 5.1: Multi-vehicle system in industrial applications: architecture

pallet) to a desired location inside the plant (e.g. a rack). The motion of the AGVs is constrained along a *roadmap*, that is a a set of virtual paths that connect every location of the plant. The fleet of AGVs is coordinated by a centralized supervisor, referred to as *control center*, that is in charge of assigning a mission to each AGV, and of coordinating the motion of the AGVs themselves. The list of missions to be accomplished is generated by the Warehouse Management System (WMS), that is integrated into the automatized logistic system of the plant. The list of missions is then sent from the WMS to the control center, for subsequent allocation to the AGVs.

5.1.1 Related Works

Several strategies can be found in the literature that consider the task allocation problem in multiple vehicle scenarios: see for instance [55] (the reader can refer to the survey papers [56] and [57]). In general, the task assignment represents an optimization problem that can be hard to solve. However it will be faced in a centralized manner through this Chapter.

The Hungarian algorithm, first presented in [3] and subsequently refined in [4], provides an optimal solution to the assignment problem in polynomial time. The

original formulation of this algorithm was developed for the *static* task assignment problem: namely, a certain number of tasks need to be assigned to a certain number of actors. However, it is worth noting that, in industrial applications, new tasks are generated as the system evolves: as an example, every time a production machine has completed the production of a batch of goods, it is necessary to remove the products and place them in the appropriate location in the warehouse. For this purpose, we need to consider a *dynamic* assignment problem: namely, it is necessary to foresee *task re-allocation*. A dynamic version of the Hungarian algorithm was presented in [58].

In multiple vehicle systems, task assignment is related to the path planning and coordination problems. In fact, once tasks have been assigned to the vehicles, subsequent path planning has to be performed to obtain the completion of the tasks themselves. The relationship between task assignment and the path planning problems have been extensively studied in [59, 60]. The main idea is that of solving an optimization problem whose solution globally optimizes the performance of the system, that is the global task completion time, taking into account both task assignment and path planning.

5.1.2 Outline

The Chapter is organized as follows. The proposed methodology is described in Section 5.2. Section 5.3 describes the implementation of the proposed methodology in a simulated environment, and presents the achieved results. Finally, Section 5.4 contains discussion of the results and conclusions.

5.2 Proposed Methodology

The proposed mission assignment methodology consists in exploiting the Hungarian Algorithm that, as is well known, represents the optimal algorithm for solving the assignment problem. Generally speaking, the Hungarian Algorithm solves the problem of assigning a certain number of *activities* to a certain number of *agents*. This assignment is based on a matrix of weights, whose element (i, j) corresponds to the cost of assigning the j-th activity to the i-th agent. The optimal assignment obtained after applying the Hungarian Algorithm has the minimum total cost among all possible choices.

In the scenario considered in this Chapter, *activities* are represented by *missions* to be accomplished, and *agents* are represented by *AGVs*. It is worth remarking that

the objective is that of increasing the overall efficiency of the system: therefore, this implies reducing the overall completion time for all the missions. Therefore, the cost for assigning each AGV to a particular mission should be proportional to the time spent by that AGV to complete that mission. Currently utilized solutions translate this idea defining the cost as a quantity that is proportional to the distance between each AGV and each mission location. In fact, assuming constant speed, travel distance is proportional to completion time. Define then $x_i(t)$ as the position of the i-th AGV at time t, and m_j as the location of the j-th mission. Therefore, the cost $c_{i,j}(t)$ is computed as follows:

$$c_{i,j}(t) = \delta(x_i(t), m_j) \tag{5.1}$$

where $\delta(\cdot)$ is the shortest path between $x_i(t)$ and m_j, computed using the Dijkstra's algorithm over the roadmap (see [61] for details). Once the roadmap has been defined, each segment s_h is characterized by its length $|s_h|$. Thus, let the shortest path between $x_i(t)$ and m_j be a sequence of $M_{i,j}$ segments, namely

$$\{s_1, \dots s_{M_{i,j}}\}$$

Subsequently:

$$\delta(x_i(t), m_j) = \sum_{h=1}^{M_{i,j}} |s_h| \tag{5.2}$$

One of the drawbacks of the definition of the cost given in (5.1) is related to the fact that assuming a constant speed for the AGVs is unrealistic, in particular in dense multi-vehicle systems. In fact, traffic jams can significantly slow down AGVs: this leads to the fact that the completion time is no longer proportional to the travel distance.

We will hereafter assume that the motion of the AGVs along the roadmap is coordinated exploiting the methodology presented in Chapter 3. In the latter the *capacity* of a sector is explained as the maximum number of AGVs that are simultaneously allowed inside the sector itself. Namely, the k-th sector is characterized by capacity $C_k > 0$.

Subsequently, based on the hierarchical division of the roadmap, it is possible to introduce the definition of traffic model.

Definition 5.1 (Traffic model)**.** *Consider a multi vehicle system characterized by:*

- *a roadmap on which the AGVs move*
- *a partitioning in N sectors*

Then, a traffic model *is a function that assigns a non-negative weight* $w_k(t)$ *that quantifies the traffic at time* t *inside the* k*-th sector.*

Considering the definition of Traffic model, the main idea in this Chapter is that of modifying the definition of the cost (5.1) to take into account the traffic itself. Specifically, let segment s_h belong to the k-th sector. Then, define the *weighted length* of s_h, namely $l_h(t)$, as a function of both the segment length $|s_h|$ and the weight $w_k(t)$, namely:

$$l_h(t) = l_h(w_k(t), |s_h|) \tag{5.3}$$

Subsequently, the cost (5.1) is defined computing the shortest path $\delta(x_i(t), m_j)$ utilizing the Dijkstra's algorithm over the roadmap, where each segment s_h is characterized by the weighted length $l_h(t)$ defined as in (5.3).

It is worth noting that modeling the traffic in a multi vehicle system is still an open problem, definition 5.1 is also used in Chapter 6 where a probabilistic dynamic model of the traffic is presented and it is used to plan the paths for the AGVs avoiding congested sectors.

5.3 Evaluation

The proposed methodology has been implemented in the same simulated environment used in Chapter 3 and through the Thesis. In particular, this methodology is validated with simulations performed on a medium size plant, characterized by:

- Number of AGVs: 30
- Number of sectors in the roadmap: 18

The traffic model utilized in the simulations is the following. We consider each sector to be characterized by the same value of capacity $C > 0$: namely, $C_k = C$, $\forall k$. Moreover, let $n_k(t)$ be the number of vehicles in the k-th sector at time t. The weight is formally formulated as follow.

$$w_k(t) = \left(\frac{n_k}{C - n_k}\right) - 1 \tag{5.4}$$

The weight function is depicted in Fig. 5.2 considering the capacity $C = 10$.

Subsequently, the weighted length l_h is computed as follows:

$$l_h = k_1 |s_h| + k_2 w_k \tag{5.5}$$

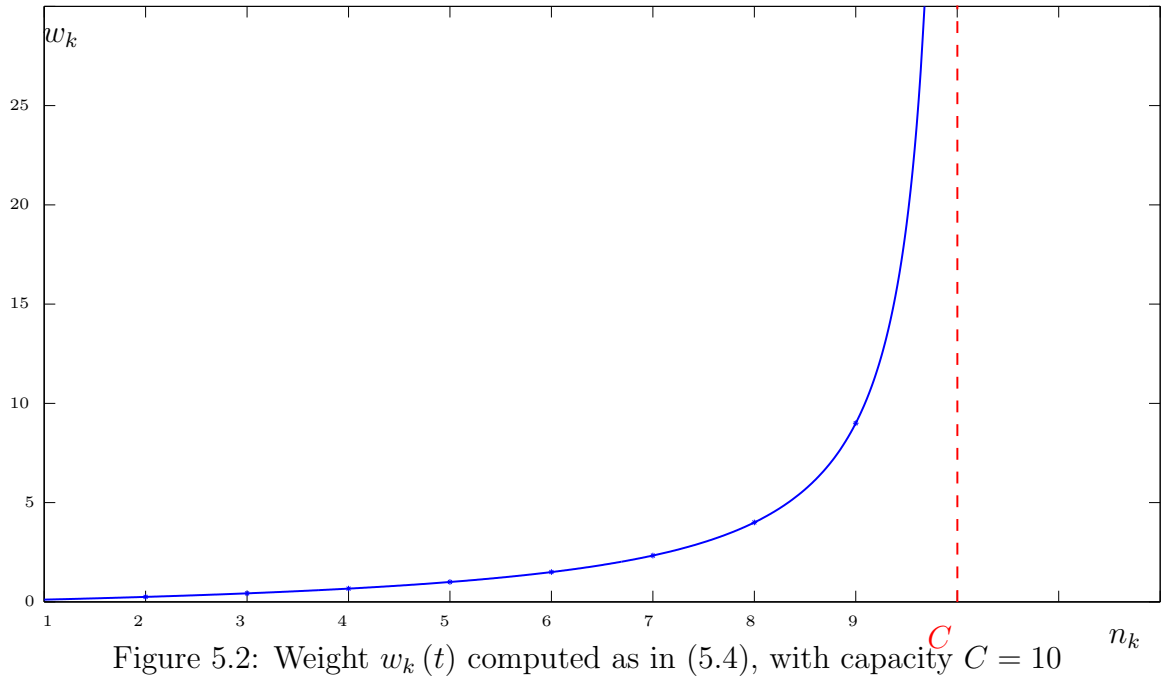


Figure 5.2: Weight $w_k(t)$ computed as in (5.4), with capacity $C = 10$

where $k_1, k_2 > 0$ are design parameters. This weighted length is the same of Eq.(3.1) in Chapter 3. Simulation results are reported hereafter for $k_1 = 1$, $k_2 = 10$, for different values of the capacity $C \in [2, \ldots, 30]$. In particular, simulations were performed randomly generating missions to be accomplished by the AGVs. Then, we measured the average number of missions accomplished per hour, and we compared it with the current result: namely, computing the weights based on path length only, that is obtained with $k_2 = 0$. Fig. 5.3 depicts the percentage increase.

5.4 Discussion and conclusion

The Chapter has presented an enhanced mission assignment methodology, for multiple AGVs used in industrial environments for logistics operations. Since traffic jams have great influence on the effectiveness of an optimal assignment, the idea of assigning missions taking into account the current traffic state has been introduced. In particular, we introduced a heuristic method that considers the weight of each path computed taking into account the state of occupation of each sector, together with the path length.

A critical design parameter is represented by the capacity of the sectors, which heavily influences the performance of the system. Very promising results were obtained for medium values of the capacity, namely $C \in (10, 15)$: in these cases, the number of missions accomplished per hour increased by more than 10%. It is worth

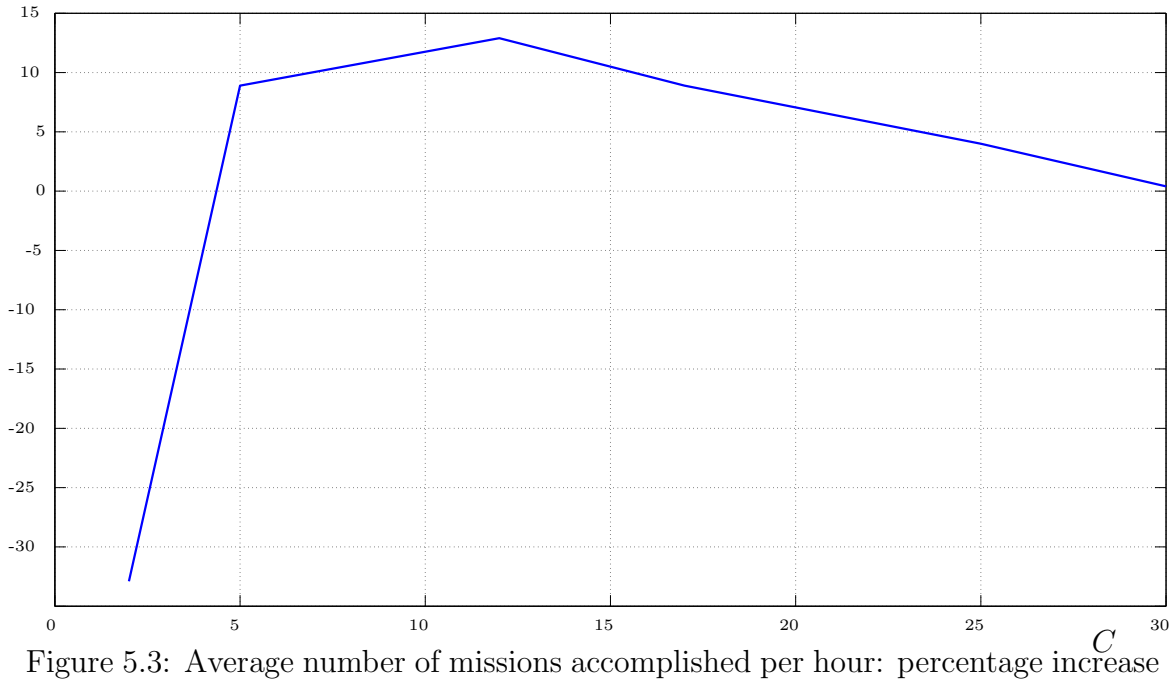


Figure 5.3: Average number of missions accomplished per hour: percentage increase with respect to the nominal case, i.e. $k_2 = 0$, for different values of the capacity C

noting that, for high value of the capacity, the performance is very similar to the nominal case. This is however expected, since it is very unlikely to have a high number of AGVs concentrated in the same sector. Therefore, traffic has only a small influence on the assignment procedure. Conversely, for very small values of the capacity (i.e. $C < 5$) we observe a decrease in the performance: this is mainly due to the fact that a sector containing a small number of AGVs are identified as heavy traffic spots, that lead to a suboptimal mission assignment.

Chapter 6

Traffic Model

In the previous Chapter, we have seen the importance of modeling the traffic in order to manage the fleet in a efficient manner. In Chapter 5 the traffic has been modeled based on the maximum number of AGVs contained in the sectors. This Chapter aims at extending the notion of traffic model. In particular, a probabilistic dynamic model is proposed which predicts and tracks the evolution of possibly congested areas. This can then be used to manage the fleet in order to avoid traffic jams. Such a model is then exploited for building a predictive planner that is embedded in the traffic manager proposed in Chapter 3 in order to explicitely consider the evolution of the traffic up to a given horizon and to increase the efficiency of the fleet of AGVs in terms of delivery time. The proposed traffic manager is also described in [62].

6.1 Introduction

In Chapter 3 a hierarchical planner is shown where the robots circulate in an environment partitioned into sectors. The sequence of sectors each AGV has to cross for reaching its destination is computed by a centralized planner, in order to guarantee high performance. Inside each sector, the AGVs are coordinated using a decentralized strategy which guarantees a simple and scalable traffic management. The granularity of the partition represents the balance between the centralized and the decentralized nature of the traffic manager. The centralized planner works on the sectors set in a Model Predictive Control (MPC) fashion. Each time an AGV enters a new sector, the central unit measures the traffic in each sector by simply counting the number of AGVs it contains. Subsequently, using the A^* algorithm [15] the sequence of sectors the AGV needs to track is recomputed considering the current traffic situation. The introduction of such a simple traffic measure has significantly increased the performance of the traffic manager, see [43] for more details.

Nevertheless, the use of a "static metric" as the one exploited in [43] and in Chapter 5 can lead to some pitfalls. In fact, it can happen that one AGV is routed towards an empty sector but that, when the AGV reaches it, the sector is full of AGVs that have reached it in the meanwhile. Thus, the AGV will take much longer than expected for crossing the sector. This is due to the fact that traffic is a highly dynamic process. In order to take a more meaningful decision on the sectors an AGV should cross, it is necessary to know the amount of AGVs in the sectors at the planning time and in a future horizon. This can be done by building a model of the traffic that cannot be deterministic, because of the unexpected events that can take place in the warehouse and because of the unpredictable negotiation time each AGV has to wait for in a sector.

6.1.1 Related Works

Understanding the evolution of the traffic is a very important issue in road design and control for urban vehicles and several modeling strategies are available in the literature (see e.g. [63]). A traffic scenario similar to the one that can be found in automatic warehouses is that of air traffic control, where some traffic models have been built for predicting the number of aircrafts that are contained in certain control areas [64]. In particular, Eulerian models, based on conservation equations on control volumes, have been successfully exploited in [65]. Probabilistic models for modeling the air traffic have also been proposed [66]. A branch of research aims at modeling the traffic by means of *queuing network models* [67, 68]. These models can manage the stochasticity of the traffic, such as the presence of unforeseen events, the delays due to congestions and they are very tractable with standard theoretical methods.

Nevertheless, at the best of the authors' knowledge, no model of the traffic of AGVs in automatic warehouses has been proposed so far and the models available from other application domains cannot be directly exploited. In fact, the air traffic models can not include unexpected stops, which are rather common in automatic warehouses. Furthermore, the AGVs flow can neither be discretized as the aircraft flow nor be modeled as a continuous flow as in urban vehicles traffic [69, 70]. On the other hand, the queuing models provide a macro-model of the traffic and this could not capture the dynamics of the local coordination among the vehicles. The exploration of queueing models for automatic warehouses is then left for future works and an Eulerian model is chosen in this work.

In this Chapter we aim at developing a probabilistic dynamic traffic model in

order to track the congested areas in the warehouse over time. Such a model is tailored on the topological structure of the warehouse and it takes into account, probabilistically, the presence of unexpected events that cause unexpected stops of the vehicles. Such a traffic model is then embedded in the traffic manager proposed in Chapter 3 in order to allow the planner to drive the AGVs along sectors with few vehicles. In this way, the time spent by each AGV inside a sector for coordinating with the other vehicles will be drastically reduced and the performance of the overall system will improve. The effectiveness of the extended traffic manager will be validated on a real warehouse scenario. In summary, the contribution of this work is twofold:

- a probabilistic traffic model for a group of AGVs travelling in a warehouse subject to unexpected events and
- a novel traffic manager, based on [43], that exploits the proposed traffic model for improving the AGVs coordination and, consequently, for increasing the efficiency of the fleet.

6.1.2 Outline

The Chapter is organized as follows: some preliminaries and the statement of the addressed problem are provided in Sec. 6.2. Sec. 6.3 formally describes the traffic model and Sec. 6.4 shows the novel planner based on the traffic model. Finally Sec. 6.5 validates the proposed results on a real warehouse and Sec. 6.6 concludes with some discussions.

6.2 Preliminaries

In this section we will briefly report the notions presented in Chapter 3 and Chapter 5, where more details can be found.

In automatic warehouses a fleet of N AGVs moves along roadmap $\mathcal{R}$, namely a set of predefined paths which the AGVs can track. For safety reasons, the roadmap is partitioned into a collection of segments $\mathcal{P} = \{p_1, \ldots, p_Q\}$, where Q is the total number of segments in $\mathcal{R}$, and it can be naturally modeled as a strongly connected graph. A path on $\mathcal{R}$ is simply a sequence of adjacent segments the AGV can track. The velocity of an AGV along a segment is constant and each segment can be occupied by one AGV at a time. Furthermore, we assume that AGVs have different pairs of initial and final configurations.

In order to implement an efficient and computationally affordable traffic manager system, in Chapter 3, the warehouse (and consequently the roadmap) has been partitioned into a set of sectors $\mathcal{S} = \{S_1, \ldots, S_P\}$, where P is the total number of sectors. An example of roadmap and of sector partitioning, as well as segments and paths, on a real warehouse is reported in Fig. 6.1.

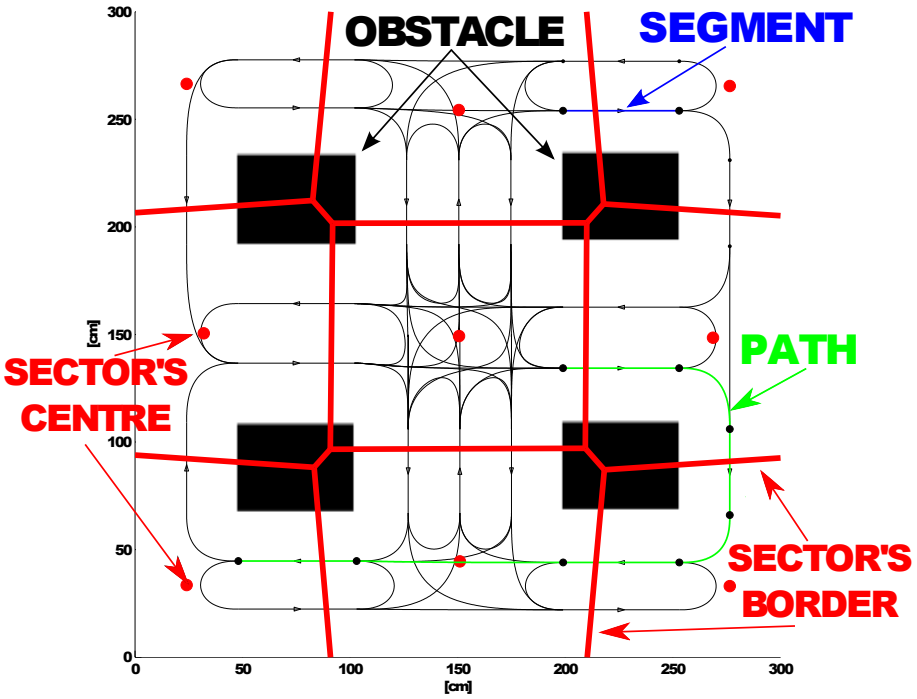


Figure 6.1: Roadmap of real environment.

The sector partition provides a high-level coarse topological representation of the roadmap that allows to abstract some properties of the fleet to make them more tractable. In particular, it is possible to associate a traffic measure to each sector rather than distribute it along the whole roadmap and, therefore, a finite state model of the traffic can be built. Formally, let $C_i \in \mathbb{R}^+$ be the capacity of S_i, i.e. the maximum number of AGVs that can be contained in S_i, and let $Y_i(k)$ be the number of AGVs travelling in S_i at time k. The higher is the density of AGVs in the sector, the bigger is the traffic in the sector. Thus, the traffic measure for sector S_i at time k is given by:

$$T_i(k) = \frac{Y_i(k)}{C_i - Y_i(k)} \tag{6.1}$$

Considering the current traffic measure, it is possible to represent the set of sectors as a weighted directed graph $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0)$ where each sector is a vertex $S_i \in \mathcal{V}_0$ and $(S_i, S_j) \in \mathcal{E}_0$ if there is a path on the roadmap completely contained in $S_i \cup S_j$,

starting from S_i and finishing in S_j. Each edge is weighted by the following time varying function:

$$\omega_{i,j}(k) = K_{ij}T_j(k) + D_{i,j} \tag{6.2}$$

where $D_{ij} \in \mathbb{R}^+$ is the Euclidean distance between the geometric centers of S_i and of S_j and $K_{ij} \in \mathbb{R}^+$ is a design parameter that allows to further weight the traffic component. The weight encodes the time necessary for traveling form S_i to S_j. The higher the traffic in S_j and the bigger the distance between the sectors, the more time it will take to travel through sector S_j.

As previously proposed, the traffic manager works on two layers. In order to deal with the highly dynamic nature of the warehouse, each time an AGV enters a new sector, the planner re-computes the sequence of sectors to be crossed considering the current traffic situation. After the planning has been done, the AGV crosses the sector it just entered in and it heads toward the first planned sector. Nevertheless, during the time requested for reaching the planned sector, the traffic situation changes and this can make the planned choice a bad choice. This happens because the planner takes a snapshot of the traffic at time k and based on that it plans the future sectors for the AGV. In order to increase the efficiency of the traffic manager we need to solve the following problem:

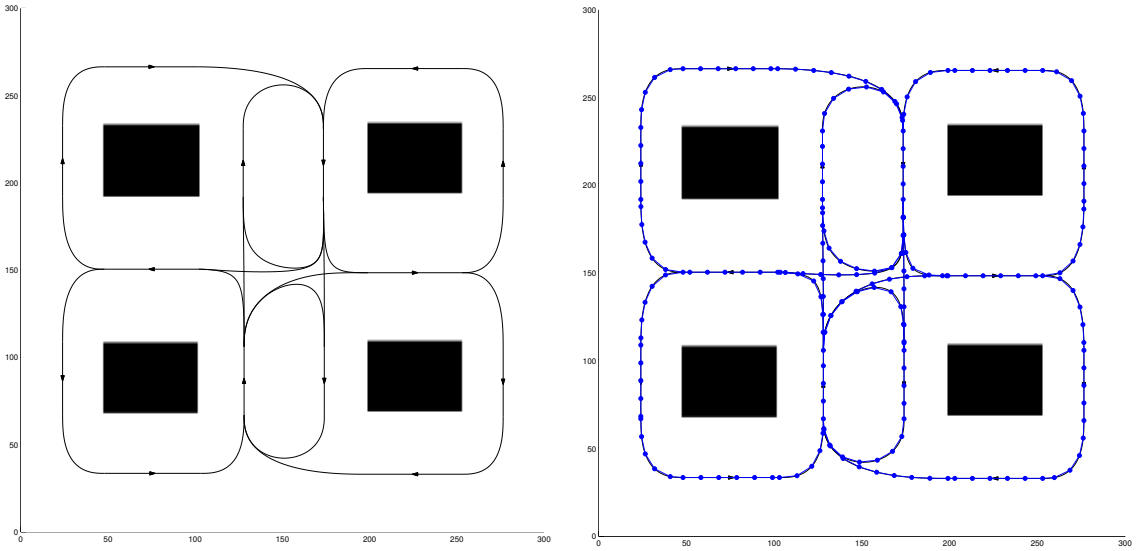
Problem 6.1. *Given an automatic warehouse partitioned into sectors, find a metric capable of tracking the dynamic evolution of the traffic of AGVs over a certain horizon and embed this information in the traffic manager proposed in Chapter 3.*

6.3 Probabilistic Traffic Model

In this section we propose a simple model that allows to track the traffic flow among the sectors obtained by the partitioning of the warehouse. In particular, exploiting the large-capacity cell transmission model [71] used for modelling air traffic flows, we build an eulerian model based on a graph-based representation of the traffic flow. Furthermore, unexpected events are managed by introducing a probabilistic term modeling the possibility that an AGV gets stuck in a segment.

Since the segments the roadmap is partitioned in are in general of different lengths, in order to build the Eulerian model, we need to further partition the roadmap in a set of *cells* $\mathcal{C} = \{c_1, \ldots, c_M\}$, where M is the total number of cells in $\mathcal{R}$. Each cell can be tracked in the same amount of time and we take the cell traveling time T_c as the unit of the discrete time set. In the following, for ease of notation, we will indicate the time instant kT_c with k. The roadmap $\mathcal{R}$ partitioned

in to cells can be represented as a graph $\mathcal{G}_c = (\mathcal{V}_c, \mathcal{E}_c)$ where the nodes are the cells, $c_i \in \mathcal{V}_c$, and where $(c_i, c_j) \in \mathcal{E}_c$ if c_i are c_j are adjacent, namely if the end point of c_i is the starting point of c_j. Since $\mathcal{R}$ can be represented as a strongly connected graph, then $\mathcal{G}_c$ is strongly connected as well. The roadmap $\mathcal{R}$ and the corresponding cell decomposition on a real warehouse are reported in Fig. 6.2.



(a) Roadmap (b) Cell Graph

Figure 6.2: Roadmap and associated graph of cells

Exploiting the conservation of flow in a cell, it is possible to relate the probability that a cell contains a vehicle at time $k+1$ to unexpected events and to the probability that a neighboring cell is containing an AGV at time k. Let $x(k) = (x_1(k), \ldots, x_M(k))^T \in \mathbb{R}^M$ be the *cell probability vector*, where $x_i(k)$ is the probability that the i^{th} cell contains an AGV at time k. At time k an AGV can either move to an adjacent cell (because of coordination reasons) or stand in the cell where it is (because of an unexpected events that prevents it from advancing). Thus, it is possible to model the evolution of the cell probability vector as:

$$x(k+1) = \mathcal{L}x(k) \tag{6.3}$$

where

$$\mathcal{L} = R_g A^T + \rho_s I. \tag{6.4}$$

The I is the M dimensional identity matrix and $A \in \mathbb{R}^{M \times M}$ is the adjacency matrix of the cell graph $\mathcal{G}_c$ defined as:

$$[A]_{i,j} = \begin{cases} 1 & \text{if } (c_i, c_j) \in \mathcal{E}_c \\ 0 & \text{otherwise} \end{cases}$$

and the diagonal matrix $R_g = diag(r_g^1, \ldots, r_g^M) \in \mathbb{R}^{M \times M}$ is such that:

$$r_g^i = \frac{\rho_g}{\sum_{j=1}^{M} A_{i,j}^T}. \tag{6.5}$$

Finally, $\rho_g, \rho_s \in [0, 1]$, with $\rho_g + \rho_s = 1$, are design parameters and they represent the probability that AGVs move to the next cell or that they stay in the current cell because of an unexpected event respectively. The parameter ρ_g is necessary because when the AGVs are executing the decentralized policy proposed in Chapter 3 it is not possible to know deterministically the sector they will choose. The matrix R_g is used for equally distributing ρ_G over the cells that can be reached at the next time step[1].

Remark 1. *In* (6.4) *we have assumed that all the AGVs have the same probability of advancing or of being subject to an unexpected event. In real warehouses the probability may change depending on the position of the cell in the warehouse. The proposed model can be easily extended to cope with this situation. In order to keep the notation simple, we consider uniform probabilities.*

In order to avoid to evaluate as congested an area with many cells with a low probability of being occupied, we build the *cell occupancy vector* as:

$$\gamma(k) = [\gamma_1(k), \ldots, \gamma_M(k)]^T \in \mathbb{R}^M, \tag{6.6}$$

where the cell probability vector has been filtered as follows:

$$\gamma_i(k) = \begin{cases} 1 & \text{if } x_i(k) \geq \epsilon \\ 0 & \text{otherwise} \end{cases} \quad \text{for } i = 1, \ldots, M \tag{6.7}$$

where the threshold ϵ is a design parameter. In this way, a cell is deemed occupied only if its probability of being occupied is big enough. Finally it is possible to evaluate the number of AGVs that are circulating in each sector by:

$$Y_j(k) = B_j \gamma(k) \quad j = 1, \ldots, P \tag{6.8}$$

where $B_j = (b_1^j, \ldots, b_M^j) \in \mathbb{R}^{1 \times M}$ is a selection vector such that

$$b_m^j = \begin{cases} 1 & \text{if } c_m \in S_j \\ 0 & \text{otherwise} \end{cases} \qquad m = 1, \ldots, M \tag{6.9}$$

It is possible to note that the threshold ϵ provides a way to manage the reliability of the model: as the time approaches infinity the reliability decreases. In particular,

[1]Since $\mathcal{G}_c$ is strongly connected, $\sum_{j=1}^{M} A_{i,j}^T \neq 0 \ \forall j$ and therefore (6.5) is well posed

the state Y_i of the sectors goes to zero for long-term predictions because of ϵ. Fig. 6.3 shows this statement. Thus, starting form the knowledge of the occupation of the cells (i.e. the position of the AGVs) at time k, it is possible to use (6.3), (6.7) and (6.8) together with (6.1) for building an estimate of the traffic contained in each cell at time in the future instants of time. The following result holds:

Theorem 6.1. *The system* (6.3) *is marginally stable.*

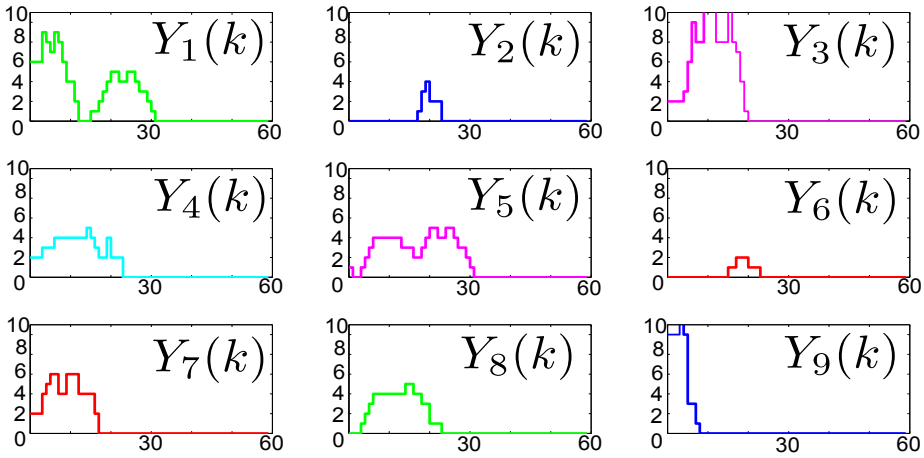
Proof. A^T is the adjacency matrix of $\mathcal{G}_c^T$, the reverse graph of $\mathcal{G}_c$ and $R_g A^T$ is the weighted adjacency matrix of $\mathcal{G}_c^T$. $\mathcal{L}$ is then the adjacency matrix of a graph $\mathcal{G}_{cs}^T$ obtained by $\mathcal{G}_c^T$ by adding a weighted self-loop on each vertex. Being $\mathcal{G}_c$ strongly connected, $\mathcal{G}_c^T$ is strongly connected [41] and, trivially, also $\mathcal{G}_{cs}^T$. The rest of the proof is an application of the Perron-Frobenius theorem. From the strong connectivity of $\mathcal{G}_{cs}^T$, we can state that $\mathcal{L}$ is irreducible, that it has a positive Perron-Frobenius eigenvalue r and that each other eigenvalue λ_i is such that $|\lambda_i| < r$. The eigenvalue r is such that $min_i \sum_{j=1}^m l_{ij} \leq r \leq max_i \sum_{j=1}^m l_{ij}$, where l_{ij} is the generic element of $\mathcal{L}$. By construction $min_i \sum_{j=1}^m l_{ij} = max_i \sum_{j=1}^m l_{ij} = \rho_g + \rho_s = 1$ and therefore $r = 1$. Thus the state matrix of the linear discrete time system in (6.3) has one simple eigenvalue on the unit circle and all the other eigenvalues inside the unit circle and therefore the system is marginally stable. □

An evidence of this claim is that the elements of $\gamma(k)$ become constant when the time goes to infinity and, consequently, the values of the traffic measure for each sector become constant. Let $\bar{Y}_i$ be the values corresponding to the number of AGVs contained in each sector without the filtering of the cell probability vector by means of the parameter ϵ. Then, Fig. 6.4 shows an example, where the values of $\bar{Y}_i$ are plotted.

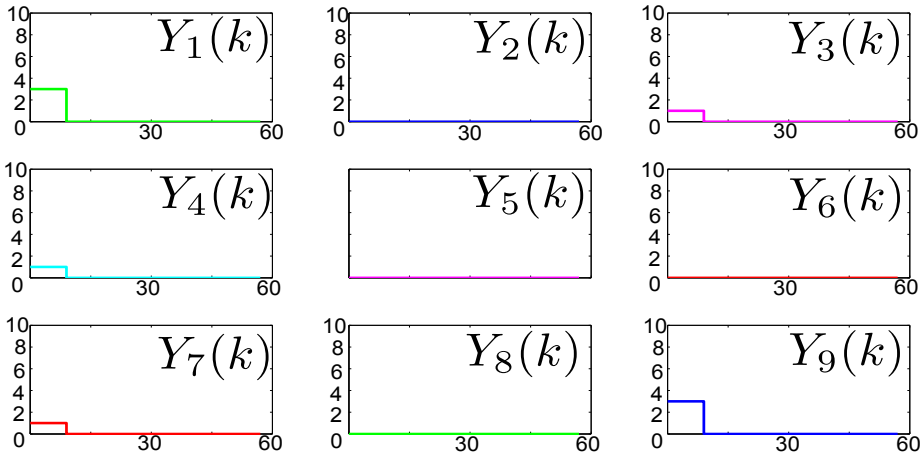
In other words, the presented model represents a Markov chain with transition matrix $\mathcal{L}$. The uncertainty of the model (the entropy of the distribution of probability) is given only by the second largest eigenvalue of $\mathcal{L}$. In particular, since $\mathcal{L}$ is a stochastic, irreducible and aperiodic matrix, the following statement holds:

$$\mathcal{L}^k = \mathbf{1}\pi^k + O(\lambda_2^k)$$

Where λ_2 is the second largest eigenvalue of $\mathcal{L}$ and π is the stationary distribution vector such that $\pi = \pi\mathcal{L}$, namely π represents the values of $\mathcal{L}$ when the time approaches infinity. The value of λ_2 is given by a non-linear function of the parameters ϵ, ρ_s and ρ_g. The parameters are empirically tuned but a more theoretical characterization is planned for future works. However, it is possible to observe that smaller values of ϵ lead to a slower convergence of the model, as shown in Fig. 6.3.



(a) $Y_j(k)$ with $\epsilon = 0.1$



(b) $Y_j(k)$ with $\epsilon = 0.5$

Figure 6.3: State $Y_j(k)$ for different values of ϵ

Thus, the model (6.3) is well posed, since the value of the prediction does not go to infinity, but the farther in time the prediction is made the less able to track the traffic the model is and the less reliable the prediction is. Thus, the traffic prediction should be made over a short-enough horizon and updated every time new data (e.g. position of the AGVs) are available.

6.4 Dynamic Traffic-Based Planner

In this section we will illustrate how to embed the dynamic traffic model proposed in Sec. 6.3 in the high-level planner of the traffic manager discussed in Sec. 6.2 for

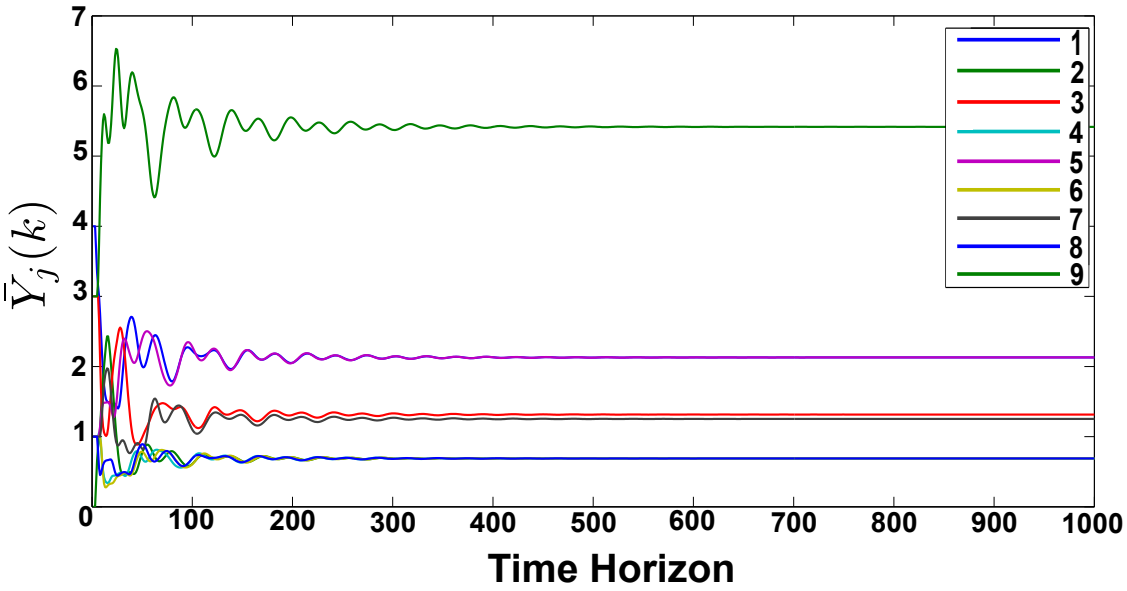


Figure 6.4: State $\bar{Y}_j(k)$ of nine sectors as long as the time goes to infinity.

replacing the static traffic metric used in Chapter 3 and in [43].

The high-level planner considers the traffic for weighting the sectors graph $\mathcal{G}_0$ at the planning times. In order to consider the time evolution of the traffic that can be provided by the traffic model, it is necessary to embed the time information into $\mathcal{G}_0$. This can be done by exploiting the time-expanded network concept.

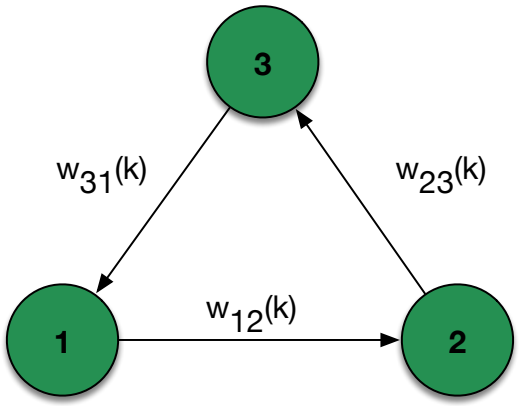
In the rest of the Chapter, we refer to the traffic manager proposed here as *Traffic-Planner*, and to the one proposed in Chapter 3 as *Basic-Planner*.

6.4.1 Time-Expanded Network

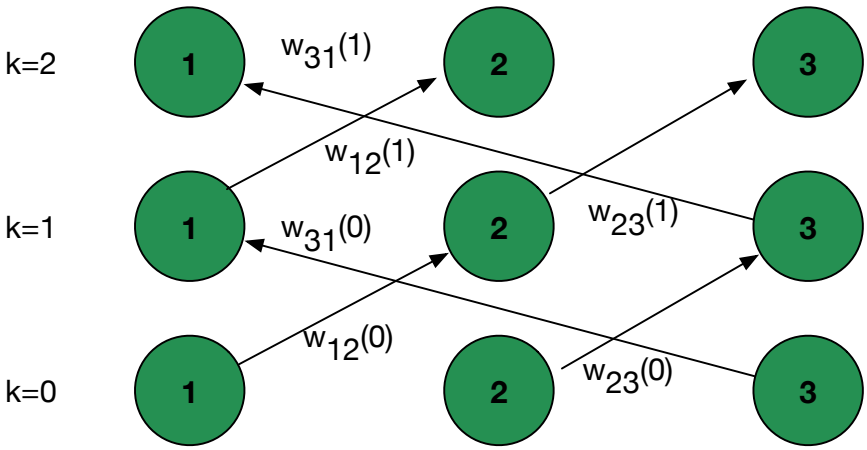
In this subsection we briefly introduce the concept of Time Expanded Network (TEN) over a discrete time interval. More details can be found in [72, 73].

Consider a weighted directed graph $G_0 = (V_0, E_0)$ where $v_i \in V_0$ and each edge $e_{ij} := (v_i, v_j) \in E_0$ is weighted with a time varying weight $w_{ij}(k)$, $k \in \mathbb{Z}^+$, which encods the cost of transitioning from v_i to v_j. We assume that each transition can be made in one time step. The finite TEN is a directed graph $G_k = (V_k, E_k)$ that is obtained by expanding G_0 over time by making a separate copy of all the nodes for each time instant $k \in \mathcal{H} = 0, 1, \ldots, H-1$, where H is the finite expansion horizon. Each node in the TEN is labeled with the time layer to which it belongs, thus every vertex of G_k is a node-time pair (v_i, k) where $v_i \in V_0$ and $k \in \mathcal{H}$. Each edge $e_{ij}^k \in E_k$ is associated to $e_{ij} \in E_0$ but it encodes the time information and it connects (v_i, k) with $(v_j, k+1)$, with $k \in \mathcal{H}$. Finally, the weight structure of G_0 is inherited by the TEN G_k and each edge $e_{ij}^k \in E_k$ is weighted by $w_{ij}(k)$. Of course, depending on

the considered time layers, the weight of the edge e_{ij}^k changes and, therefore, using TEN it is possible to consider explicitly the time evolution of the weights of the graph over the expansion horizon. One of the main advantages of the use of TEN is the possibility of solving shortest path problems in a dynamic graph using static shortest path algorithms on its associated TEN. An example of TEN is shown in Fig. 6.5 where a TEN (Fig. 6.5b) is associated to a graph G_0 (Fig. 6.5a).



(a) Basic network graph, the arc weight is shown with an integer.



(b) Time-Expanded network

Figure 6.5: Original graph and associated time expanded graph with $H = 3$.

6.4.2 Traffic-based Planner

Consider the graph $\mathcal{G}_0$, where each node is a sector S_i and edges represent a roadmap interconnection between the joined sector. Each edge is weighted by the traffic metric reported in (6.1). Each time an AGV exits a sector, the *Basic-Planner* plans

the path over $\mathcal{G}_0$ for a finite horizon $\mathcal{H} = \{k, \ldots, k + H - 1\}$. The planner works in a model predictive control fashion, namely the whole path along the sectors is computed but only the first step is actually executed by the AGVs. The path is then re-computed whenever an AGV exits a sector.

The traffic model reported in (6.3)-(6.8) allows to compute the number of AGVs contained in each sector for the planning and, therefore, the evolution of the traffic metric $T(k)$, and consequently of the edge weights $\omega_{i,j}(k)$, for $k \in \mathcal{H}$. In order to encode such a dynamic traffic information in the planner, we build the TEN $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$ associated to $\mathcal{G}_0$ for the time horizon $\mathcal{H}$, where the weight associated to each edge $\mathcal{E}_k \ni e_{ij}^k$ is $\omega_{ij}(k)$. In this way, the TEN $\mathcal{G}_k$ encodes the dynamic traffic information obtained thanks to the traffic model in a static graph. Using the same A^* [15] strategy of the *Basic-Planner* for planning on the TEN $\mathcal{G}_k$, it is now possible to obtain more meaningful high-level paths for the AGVs, which take into account how the traffic of the vehicles evolves in the warehouse and that avoid to enter in too crowded sectors, something that can happen using the traffic metric used in the *Basic-Planner*. The strategy for coordinating the AGVs when they are traveling inside a sector is the same decentralized protocol based on priorities adopted in Chapter 3. In Fig. 6.6 a portion of the TEN exploited for the traffic management of a real warehouse is shown. The blue lines are the edges of the TEN.

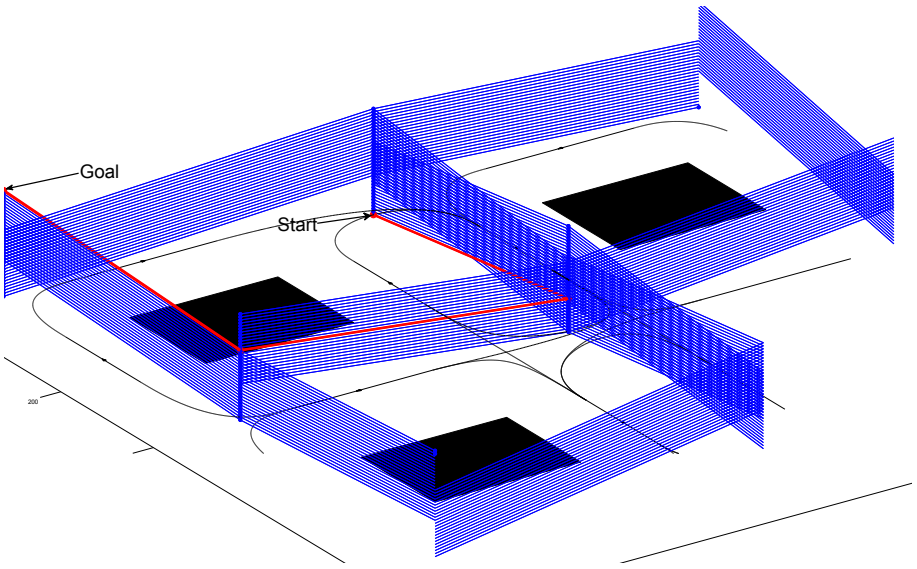


Figure 6.6: The TEN used in the analyzed scenario.

As noticed in Sec. 6.3, the information provided by the traffic model is the less reliable the more into the future the prediction is made and this could lead to wrong choices of the paths. Nevertheless, the planner replans the path for the AGV each

time that the vehicle enters in a new sector and, therefore, the AGV travels along only to the first sector of the planned path, namely the one which comes from the most reliable traffic information.

In general the TEN can be very big in terms of nodes and edges, and the graph search process can be slow. Nevertheless, the planner works on the high-level representation of the roadmap which can be made coarse enough to avoid the TEN dimensional explosion.

6.5 Validation

In order to validate the traffic model proposed in Sec. 6.3 and the traffic control strategy derived by the *Basic-Planner* by embedding the planner proposed in Sec. 6.4, we made several simulations in Matlab considering real warehouses plants.

The simulations are performed on the real scenario shown in Fig. 6.2a. The proposed traffic manager is compared with the coordination strategy presented in previously which does not exploit a traffic model in the high-level planning. The decentralized coordination inside each sector is the same for the two traffic managers, what is different is the high level planner.

The comparison has been carried out analyzing the total arrival time and the ratio between the time the AGVs get stuck and the time the AGVs are in movement. The simulations are performed according with the following conditions:

- number of AGVs $= [3, 5, 10, 15]$
- 25 runs for each configurations
- same tasks (randomly generated in each run) and initial positions for the compared methods
- same low level decentralized strategy applied to both the methods
- number of sectors: $P = 9$
- number of cells: $M = 231$
- design parameter of (6.2): $K_{ij} = 100 \ \forall i, j$
- sector's capacity $C = [3, 1, 2, 1, 3, 1, 2, 1, 8]$
- probability parameters: $\rho_g = 0.8$, $\rho_s = 0.2$

- filter threshold parameter $\epsilon = 0.5$

The maximum number of AGVs, i.e. the capacity, of each sector C_j is computed offline. The results are shown in Fig. 6.7 where the improvement of the *Traffic-Planner* with respect to the *Basic-Planner* is shown in percentage. The variance is also highlighted. It is possible to note that the proposed method performs better in terms of total arrival time than the previous one: the improvement is up to 22%. The average values obtained in the different runs when using the Traffic-Planner are always lower than the corresponding ones when using the Basic-Planner (that are already better than the ones currently achieved in the industrial practice [51]). The second term of comparison is the ratio between the time the AGVs get stuck and the time the AGVs move and the result is shown in Fig. 6.9. It is possible to note that the *Traffic-Planner* performs better as the number of AGVs increases with respect to the *Basic-Planner*.

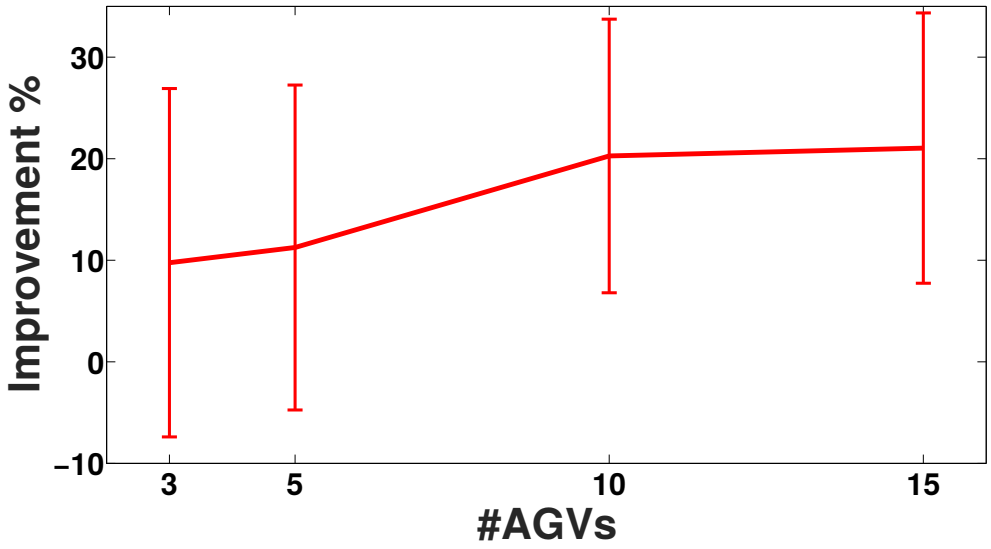


Figure 6.7: Traffic-Planer's improvement (percentage).

The simulation data show that the performance of the proposed strategy are always greater or equal than the ones that can be obtained using the strategy presented in Chapter 3, see e.g. Fig. 6.8. This is due to the fact that the first sector planned by the *Traffic-Planner* cannot be worse, in terms of traffic, than the one obtained by the *Basic-Planner*. In other words, it is possible to say that the performance of the *Traffic-Planner* are lower bounded by the performance of the *Basic-Planner*. The result data are also evaluated in terms of significance. Table 6.1 shows the statistical evaluation where $h = 1$ indicates a rejection of the null hypothesis at the

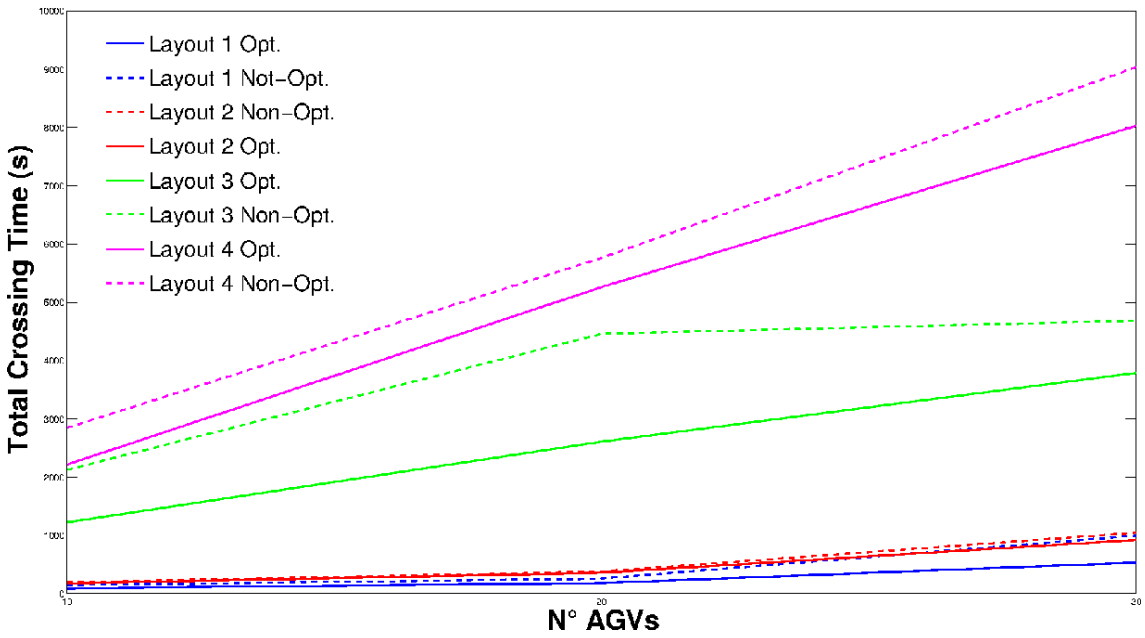


Figure 6.8: The evaluation data for the 5 AGVs simulation.

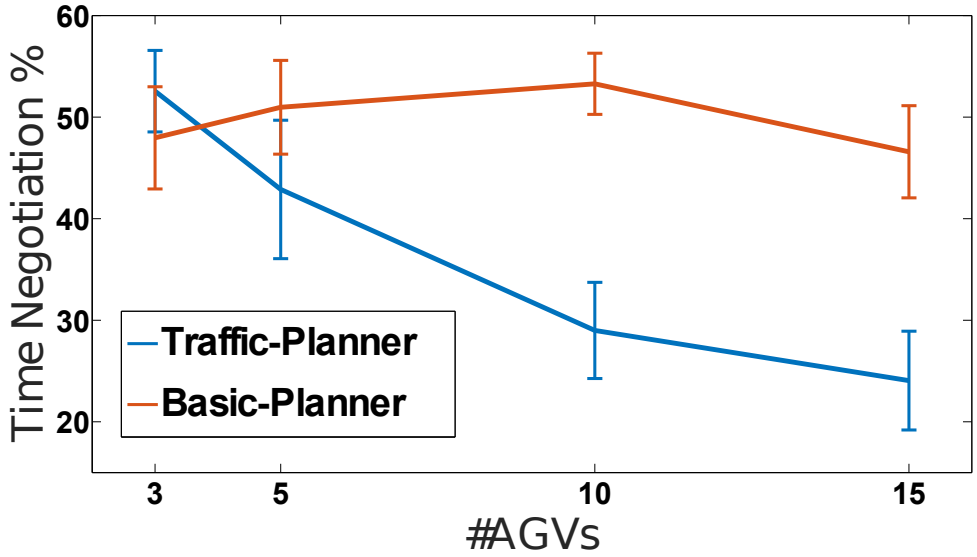


Figure 6.9: Negotiation time with respect to the time where the AGVs are in movement.

5% significance level and $h = 0$ indicates a failure to reject the null hypothesis at the 5% significance level.

#AGV	h	p-value
3	1	0.0296
5	1	0.0417
10	1	0.0150
15	1	0.0265

Table 6.1: Statistical validity of the comparison

6.6 Conclusion

In this Chapter we have proposed a probabilistic model for representing the traffic of a fleet of AGVs moving in an automatic warehouse and subject to unexpected stops due to unforeseen events (e.g. interaction with human guided forklifts). The model has then been embedded in the two layer traffic model proposed in Chapter 3 for improving the performance of the high level planner. Extensive simulations on a real warehouse have proven that there is a significant improvement in terms of delivery time when using the proposed traffic manager. The information about the destination of the AGVs is not included in the model in order to keep the model as simple as possible, and then in order to limit the complexity. However, future work aims at increasing the reliability of the traffic model by adding the information about the origin and the target position of the AGVs Furthermore, current work regards the analysis of the performance in a more rigorous manner.

Chapter 7

Concluding remarks

This thesis is the result of three years devoted to the research of a new strategy for coordinating multiple AGVs by considering and optimizing the overall system. The focus of the thesis has been then the development of an ensemble strategy for coordinating, optimizing and modeling an AGV system in an industrial environment. Several works focusing on multi-robot coordination can be found in literature, but the problem is always faced by considering only one aspect, e.g. the path planning mechanism. The general contribution of this work is then a different approach for multi-robot system, and in particular for AGV system. The problem is treated by considering several aspects of the AGV scenario, e.g. the roadmap and the coordination algorithm, and thus the system is faced in a holistic manner.

The main results are summarized hereafter. **Chapter 2** first describes an algorithm for the automatic generation of a roadmap for AGV system. Such a roadmap is designed in order to maximize the redundancy, the coverage and the connectivity. The core of this algorithm is the assignment of the directions to the roads (2.7). In particular, the directions can be assigned by solving an optimization problem where the maximum flow or the algebraic connectivity have be maximized (see A).

Moreover, the roadmap is modeled in such a way that the coordination algorithm detailed in **Chapter 3** works better on those kind of roadmaps. In that chapter, the hierarchical 2-layers architecture is described and a the hierarchical planner on the topological layer and on the roadmap layer is introduced.

An improved algorithm for coordinating a fleet of AGVs (one the roadmap layer) is explained in **Chapter 4**. The main contribution is the formulation of the Multi-AGV coordination problem 4.1 in a quadratic programming optimization problem. The QP problem (4.14) can be solved very efficiently and the results prove that the time wasted waiting is minimized while the total throughput is maximized.

In the last two chapters, a model of the traffic is introduced. In particular, in

Chapter 5 a static model is used for a specific application: the dynamic mission assignment. The missions are then assigned to the AGVs by considering the current status of the traffic. Simulations have shown that such a model clearly improves the number of missions per hour of the fleet.

Finally, a complete dynamic model of the traffic is validated in **Chapter 6** where the traffic is modeled in a probabilist manner. This model is then used to predict the status of the traffic in the warehouse. A *Traffic Planner* coordinates the fleet by considering the prediction of the traffic. Simulations have shown that modeling the traffic is a crucial aspect of AGV systems, and the provided model can be used to improve efficiently the performance of the whole system.

Thus, for a final remark, this Thesis aims at being a reference for the management of AGV systems under a new prospective. The focus has to be given to the whole system which can has to be modeled by using ensemble or holistic approaches.

Appendix A

Algebraic Graph Theory

In this Chapter some of the main notions on graph theory used in the Thesis are reported. For more exhaustive information, the reader can see for instance [41].

Generally speaking, a graph $\mathcal{G}$ is represented of an ordered pair $\mathcal{G} = (\mathsf{V}, \mathsf{E})$ comprising a set V of N vertices or nodes together with a set $\mathsf{E} \subseteq \mathsf{V} \times \mathsf{V}$ of edges or arcs.

Two different classes of graphs may be adopted: directed graphs and undirected graphs.

- In an *undirected graph* the connection among two nodes is bidirectional. Namely, for every couple of nodes i and j, if the $i \to j$ edge exists, then the $j \to i$ edge exists as well. Undirected graphs are thus usually exploited to model explicit bidirectional connections.

- In a *directed graph* or digraph, each edge has a direction associated with it, namely the edge $i \to j$ is different than the edge $j \to i$.

Pictorially, Fig. A.1 shows an example.

A *weighted directed graph* is a directed graph with weights assigned to its edges. In the context of graph theory, a weighted directed graph is often called a network.

A directed graph is *strongly connected* if it contains a directed path that connects every pair of nodes of the graph. A node of a graph is defined *balanced* if its out–degree is equal to its in–degree, that is the sum of the incoming (possibly weighted) edges equals the sum of the outgoing (possibly weighted) edges. A *balanced graph* is defined as a graph whose nodes are all balanced.

A communication graph can be described by means of the adjacency matrix $\mathcal{A} \in \mathbb{R}^{N \times N}$. Let $\mathcal{N}_i$ be the neighborhood of the i–th node. Each element a_{ij} of $\mathcal{A}$ is defined as the weight of the edge between the i–th and the j–th node, and is a positive number if $j \in \mathcal{N}_i$, zero otherwise.

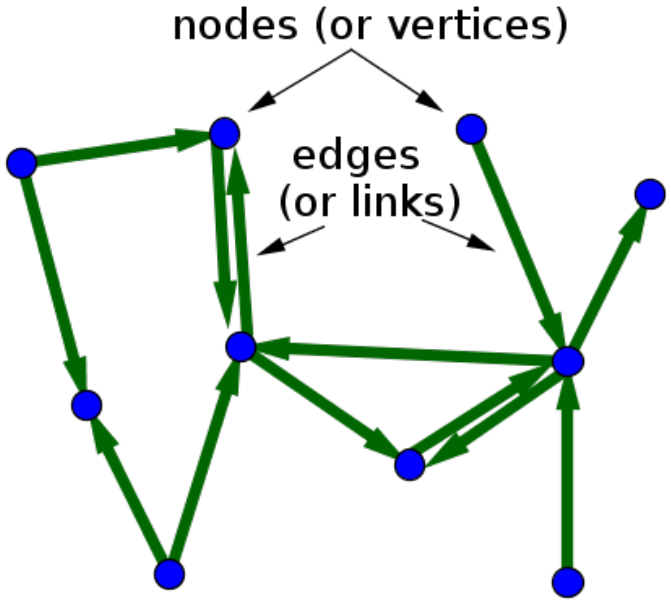


Figure A.1: Definition of directed graph

The degree matrix of the graph is defined as:

$$\mathcal{D} = \text{diag}\left(\{d_i\}\right), \tag{A.1}$$

where d_i is the out–degree of the i–th node of the graph, that is $d_i = \sum_{j=1}^{N} a_{ij}$. The (weighted) Laplacian matrix of the graph is defined as:

$$\mathcal{L} = \mathcal{D} - \mathcal{A}. \tag{A.2}$$

Let M be the number of edges and let $\mathbf{w} = [\mathbf{w}_1, \cdots, \mathbf{w}_M] \in \mathbb{R}^M$ be the weights associated to each edge, then the weight matrix $\mathcal{W} \in \mathbb{R}^{M \times M}$ is defined as $\mathcal{W} = diag(\mathbf{w})$. The Laplacian matrix $\mathcal{L}$ of the directed graph associated with the weight matrix $\mathcal{W}$ can be defined as follows:

$$\mathcal{L}_{\mathbf{w}} = \mathcal{I} \cdot \mathcal{W} \cdot \mathcal{I}^T \tag{A.3}$$

Where $\mathcal{I} \in \mathbb{R}^{N \times M}$ is the Incidence matrix. The incidence matrix $\mathcal{I} \in \mathbb{R}^{N \times M}$ is defined as a matrix whose (i,k)-th element $i_{i,k}$ is:

$$i_{i,k} = \begin{cases} -1 & \text{if } v_i \text{ is the head of } e_k \\ 1 & \text{if } v_i \text{ is the tail of } e_k \\ 0 & \text{othrwise} \end{cases} \tag{A.4}$$

The unweighted Laplacian matrix $\mathcal{L}_*$ is a special case of Laplacian matrix, where all non–zero entries of the adjacency matrix are equal to one and $\mathcal{L}_* = \mathcal{I} \cdot \mathcal{I}^T$.

The Laplacian matrix exhibits some remarkable properties [41, 74]:

1. Let $\mathbf{1}$ be the column vector of all ones. Since the row sum of the Laplacian matrix is equal to zero, then $\mathcal{L}\mathbf{1} = 0$.

2. Let $\lambda_i^{\mathcal{G}}$, $i = 1, \ldots, N$ be the eigenvalues of the Laplacian matrix of the graph $\mathcal{G}$. The eigenvalues can be ordered such that $0 = \left|\lambda_1^{\mathcal{G}}\right| \leq \left|\lambda_2^{\mathcal{G}}\right| \leq \ldots \leq \left|\lambda_N^{\mathcal{G}}\right|$

3. For a strongly connected digraph, $\Re\left\{\lambda_i^{\mathcal{G}}\right\} \geq 0 \ \forall i = 1, \ldots, N$ where $\Re\{\cdot\}$ defines the real part of a complex number.

4. In a balanced digraph, the column sum of the Laplacian matrix is equal to zero as well. Then, $\mathbf{1}^T\mathcal{L} = 0$.

5. If a graph is strongly connected, then $\Re\left\{\lambda_2^{\mathcal{G}}\right\} > 0$. In case the graph is balanced, then $\Re\left\{\lambda_2^{\mathcal{G}}\right\} > 0$ if and only if the graph is strongly connected: then, as in the undirected graph case, we refer to $\lambda_2^{\mathcal{G}}$ as the algebraic connectivity of the graph.

6. The value of $\Re\left\{\lambda_2^{\mathcal{G}}\right\}$ increases as the number of edges and with increasing the weight of the edges.

It is worth noting that the higher the value of $\Re\left\{\lambda_2^{\mathcal{G}}\right\}$ the higher the efficiency of the communication graph in solving distributed computation problems [74].

Often a graph is considered a flow network (also known as a transportation network), namely a directed graph where each edge has a capacity and each edge receives a flow (see Fig. A.2). The amount of flow on an edge cannot exceed the capacity of the edge. A flow must satisfy the restriction that the amount of flow into a node equals the amount of flow out of it, unless it is a source, which has only outgoing flow, or sink, which has only incoming flow. A network can be used to model traffic in a road system, circulation with demands, fluids in pipes, currents in an electrical circuit, or anything similar in which something travels through a network of nodes. Given two nodes of a graph, the *maximum flow* is a scalar value that quantifies the capacity of the links between those two nodes. The capacity of an edge is a mapping $c : \mathsf{E} \rightarrow \mathcal{R}^+$, denoted by $c(u, v)$. It represents the maximum amount of flow that can pass through an edge.

A *flow* is a mapping $F : \mathsf{E} \rightarrow \mathcal{R}^+$, denoted by $F(u, v)$, subjected to the following constraints:

- capacity constraint: $\forall (u, v) \in \mathsf{E} \ \ F(u, v) \leq c(u, v)$
- conservation of flows: $\forall v \in \mathsf{V}, \sum_{u:(u,v)\in\mathsf{E}} F(u, v) = \sum_{u:(v,u)\in\mathsf{E}} F(v, u)$

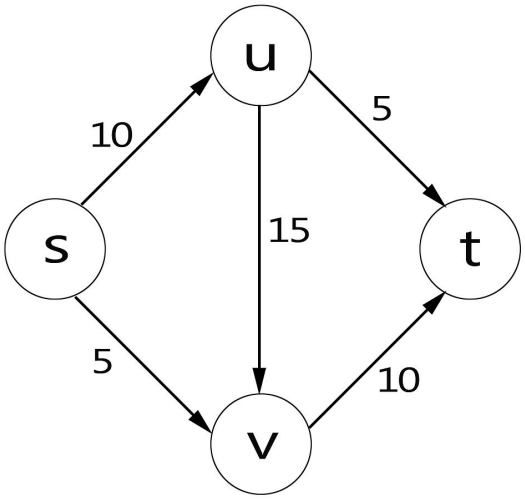


Figure A.2: A flow network, with source s and sink t. The numbers next to the edge are the capacities.

The value of the flow is defined by $|F| = \sum_{v:(s,v)\in E} F(s,v)$, where s is the source of V. It represents the amount of flow passing from the source to the sink. If the graph represents a roadmap, where a fleet of AGVs are travelling, the maximum flow can be seen as the maximum material flow, namely the quantity of goods, between two areas of the warehouse (e.g. from the production area to the delivery area). Several algorithms can be found in literature to solve such a problem. In particular, in this thesis the Edmonds–Karp algorithm [75], that is known to provide a solution with good computational performances, has been used. In particular, this algorithm seeks for the maximum flow between a source node and a sink node in $O(NM^2)$ time, where N is the number of nodes and M is the number of edges of the graph.

List of Figures

List of Author's Publications

[1] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **Towards Decentralized Coordination of Multi Robot Systems in Industrial Environments: a Hierarchical Traffic Control Strategy**. In *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2013.

[2] L. Sabattini, V. Digani, C. Secchi, G. Cotena, D. Ronzoni, M. Foppoli, and F. Oleari. **Technological Roadmap to Boost the Introduction of AGVs in Industrial Applications**. In *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2013.

[3] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **Hierarchical Traffic Control for Partially Decentralized Coordination of Multi AGV Systems in Industrial Environments**. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[4] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **Decentralized coordination enhanced by centralized information: multiple AGVs in industrial application**. In *Workshop "On the centrality of decentralization in multi-robot systems: holy grail or false idol?" Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[5] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **An Automatic Approach for the Generation of the Roadmap for Multi-AGV Systems in an Industrial Environment**. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[6] V. Digani, F. Caramaschi, L. Sabattini, C. Secchi, and C. Fantuzzi. **Obstacle avoidance for industrial AGVs**. In *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on*, pages 227–232, Sept 2014.

[7] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **Ensemble Coordination Approach in Multi-AGV Systems Applied to Industrial Warehouses**. *Automation Science and Engineering, IEEE Transactions on*, **12**(3):922–934, July 2015.

[8] Valerio Digani, M.Ani Hsieh, Lorenzo Sabattini, and Cristian Secchi. **A Quadratic Programming Approach for Coordinating Multi-AGV Systems**. In *Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE), 2015. Gothenburg, Sweden*, August 2015.

[9] Lorenzo Sabattini, Valerio Digani, Matteo Lucchi, Cristian Secchi, and Cesare Fantuzzi. **Mission Assignment for Multi-Vehicle Systems in Industrial Environments**. In *Proceedings of the IFAC Symposium on Robot Control (SYROCO). Salvador, Brazil*, August 2015.

[10] F. Oleari, M. Magnani, D. Ronzoni, L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, and C. Fantuzzi. **Improving AGV systems: Integration of advanced sensing and control technologies**. In *Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on*, pages 257–262, Sept 2015.

[11] E. Cardarelli, L. Sabattini, V. Digani, C. Secchi, and C. Fantuzzi. **Interacting with a multi AGV system**. In *Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on*, pages 263–267, Sept 2015.

[12] L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, and C. Fantuzzi. **Multi AGV Systems in Shared Industrial Environments: Advanced Sensing and Control Techniques for Enhanced Safety and Improved Efficiency**. In *Workshop on "Autonomous Industrial Vehicles: From the Laboratory to the Factory Floor", Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[13] L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, C. Fantuzzi, and K. Fuerstenberg. **Advanced sensing and control techniques for multi AGV systems in shared industrial environments**. In *Emerging Technologies Factory Automation (ETFA), 2015 IEEE 20th Conference on*, pages 1–7, Sept 2015.

[14] V. Digani, L. Sabattini, and C. Secchi. **A Probabilistic Eulerian Traffic Model for the Coordination of Multiple AGVs in Automatic Warehouses**. *Robotics and Automation Letters, IEEE*, **1**(1):26–32, Jan 2016.

[15] L. Sabattini, V. Digani, C. Secchi, and C. Fantuzzi. **Hierarchical Coordination Strategy for Multi-AGV Systems Based on Dynamic Geodesic Environment Partitioning**. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2016. submitted.

[16] L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, and C. Fantuzzi. **Multi AGV Systems in Shared Industrial Environments: Advanced Sensing and Control Techniques for Enhanced Safety and Improved Efficiency**. *in "STP1594 - Autonomous Industrial Vehicles: From the Laboratory to the Factory Floor", ASTM*, 2015. in press.

[17] V. Digani, L. Sabattini, M. A. Hsieh and C. Secchi. **Coordination of multiple AGVs: a Quadratic Optimization Method**. *Automation Science and Engineering, IEEE Transactions on*, submitted.

Bibliography

[1] T. Tsumura. **AGV in Japan-recent trends of advanced research, development, and industrial applications**. In *Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. Proceedings of the IEEE/RSJ/GI International Conference on*, **3**, pages 1477–1484 vol.3, Sep 1994.

[2] S Berman and Y Edan. **Decentralized autonomous AGV system for material handling**. *International Journal of Production Research*, **40**(15):3995–4006, 2002.

[3] Harold W Kuhn. **The Hungarian method for the assignment problem**. *Naval research logistics quarterly*, **2**(1-2):83–97, 1955.

[4] James Munkres. **Algorithms for the assignment and transportation problems**. *Journal of the Society for Industrial & Applied Mathematics*, **5**(1):32–38, 1957.

[5] Lucia Pallottino, Vincenzo G. Scordio, Antonio Bicchi, and Emilio Frazzoli. **Decentralized Cooperative Policy for Conflict Resolution in Multivehicle Systems**. *IEEE Transactions on Robotics*, **23**(6):1170–1183, December 2007.

[6] A. Farinelli, L. Iocchi, and D. Nardi. **Multirobot systems: a classification focused on coordination**. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, **34**(5):2015–2028, Oct 2004.

[7] Lynne E Parker. **Path planning and motion coordination in multiple mobile robot teams**. *Encyclopedia of Complexity and System Science. Springer, Heidelberg*, 2009.

[8] R. Olmi, C. Secchi, and C. Fantuzzi. **Coordination of industrial AGVs**. *International Journal of Vehicle Autonomous Systems*, **9**(1):5–25, 2011.

[9] Thierry Simeon, Stephane Leroy, and J-P Laumond. **Path coordination for multiple mobile robots: A resolution-complete algorithm**. *IEEE Transactions on Robotics and Automation*, **18**(1):42–49, 2002.

[10] Durdana Habib, Habibullah Jamal, and Shoab A Khan. **Employing Multiple Unmanned Aerial Vehicles for Co-Operative Path Planning**. *International Journal of Advanced Robotic Systems*, **10**, 2013.

[11] Kun Zheng, Dunbing Tang, Wenbin Gu, and Min Dai. **Distributed control of multi-AGV system based on regional control model**. *Production Engineering*, pages 1–9, 2013.

[12] M.P. Fanti, A.M. Mangini, G. Pedroncelli, and W. Ukovich. **Decentralized deadlock-free control for AGV systems**. In *American Control Conference (ACC), 2015*, pages 2414–2419, July 2015.

[13] Sergio Manca, Adriano Fagiolini, and Lucia Pallottino. **Decentralized coordination system for multiple AGVs in a structured environment**. In *18th World Congress of the International Federation of Automatic Control (IFAC 2011)*, **18**, pages 6005–6010, 2011.

[14] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. **Probabilistic roadmaps for path planning in high-dimensional configuration spaces**. *IEEE Transactions on Robotics and Automation*, **12**(4):566–580, Aug 1996.

[15] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006.

[16] Jingjin Yu and S LaValle. **Time optimal multi-agent path planning on graphs**. In *The First AAAI Workshop on Multiagent Pathfinding*, 2012.

[17] Jean-Claude Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[18] Jur P van den Berg, Dennis Nieuwenhuisen, Léonard Jaillet, and Mark H Overmars. **Creating robust roadmaps for motion planning in changing environments**. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005 (IROS 2005).*, pages 1053–1059. IEEE, 2005.

[19] Roland Geraerts and Mark H. Overmars. **A Comparative Study of Probabilistic Roadmap Planners**. In *Algorithmic Foundations of Robotics V*, pages 43–58. Springer, 2004.

[20] C Nissoux, T Siméon, and J-P Laumond. **Visibility based probabilistic roadmaps**. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 1999 (IROS99)*, **3**, pages 1316–1321. IEEE, 1999.

[21] Roland Geraerts and Mark H Overmars. **Reachability-based analysis for probabilistic roadmap planners**. *Robotics and Autonomous Systems*, **55**(11):824–836, 2007.

[22] Roland Geraerts and Mark H Overmars. **Creating high-quality paths for motion planning**. *The International Journal of Robotics Research*, **26**(8):845–863, 2007.

[23] Hyeong In Choi, Sung Woo Choi, and Hwan Pyo Moon. **Mathematical theory of medial axis transform**. *pacific journal of mathematics*, **181**(1):57–88, 1997.

[24] Alexander Kleiner, Dali Sun, and Daniel Meyer-Delius. **Armo: Adaptive road map optimization for large robot teams**. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2011*, pages 3276–3282. IEEE, 2011.

[25] Avneesh Sud, Russell Gayle, Erik Andersen, Stephen Guy, Ming Lin, and Dinesh Manocha. **Real-time navigation of independent agents using adaptive roadmaps**. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, pages 99–106. ACM, 2007.

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

[27] Peter R Wurman, Raffaello D'Andrea, and Mick Mountz. **Coordinating hundreds of cooperative, autonomous vehicles in warehouses**. *AI magazine*, **29**(1):9, 2008.

[28] L. Sabattini, V. Digani, C. Secchi, G. Cotena, D. Ronzoni, M. Foppoli, and F. Oleari. **Technological Roadmap to Boost the Introduction of AGVs in Industrial Applications**. In *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2013.

[29] Noa Agmon, Noam Hazon, and Gal A Kaminka. **The giving tree: constructing trees for efficient offline and online multi-robot coverage**. *Annals of Mathematics and Artificial Intelligence*, **52**(2-4):143–168, 2008.

[30] Zheng Sun, David Hsu, Tingting Jiang, Hanna Kurniawati, and John H Reif. **Narrow passage sampling for probabilistic roadmap planning**. *IEEE Transactions on Robotics*, **21**(6):1105–1115, 2005.

[31] Roland Geraerts and Mark H. Overmars. **Sampling and node adding in probabilistic roadmap planners**, 2006.

[32] Leena Lulu and Ashraf Elnagar. **A comparative study between visibility-based roadmap path planning algorithms**. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005 (IROS 2005).*, pages 3263–3268. IEEE, 2005.

[33] Roland Geraerts and Mark H. Overmars. **Creating High-quality Roadmaps for Motion Planning in Virtual Environments**. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4355–4361, 2006.

[34] Mark H. Overmars. **Creating small roadmaps for solving motion planning problems**. In *In IEEE International Conference on Methods, Models in Automation and Robotics*, pages 531–536, 2005.

[35] Jingjin Yu and Steven M LaValle. **Multi-agent path planning and network flow**. In *Algorithmic Foundations of Robotics X*, pages 157–173. Springer, 2013.

[36] Ioannis Karamouzas, Roland Geraerts, and A Frank van der Stappen. **Space-time group motion planning**. In *Algorithmic Foundations of Robotics X*, pages 227–243. Springer, 2013.

[37] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **Hierarchical Traffic Control for Partially Decentralized Coordination of Multi AGV Systems in Industrial Environments**. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[38] Yu-Hua Lee and Shi-Jinn Horng. **Optimal computing the chessboard distance transform on parallel processing systems**. *Computer Vision and Image Understanding*, **73**(3):374–390, 1999.

[39] Yu-Hua Lee, Shi-Jinn Horng, Tzong-Wann Kao, and Shung-Shing Lee. **Optimal parallel algorithms for computing the chessboard distance transform and the medial axis transform on RAP**. In *Proceedings of Second International Symposium on Parallel Architectures, Algorithms, and Networks, 1996.*, pages 22–28. IEEE, 1996.

[40] Toyofumi Saito and Jun-Ichiro Toriwaki. **New algorithms for euclidean distance transformation of an n-dimensional digitized picture with applications**. *Pattern recognition*, **27**(11):1551–1565, 1994.

[41] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, 2001.

[42] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **An Automatic Approach for the Generation of the Roadmap for Multi-AGV Systems in an Industrial Environment**. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[43] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **Ensemble Coordination Approach in Multi-AGV Systems Applied to Industrial Warehouses**. *Automation Science and Engineering, IEEE Transactions on*, **12**(3):922–934, July 2015.

[44] L. Sabattini, C. Secchi, and N. Chopra. **Decentralized Control for Maintenance of Strong Connectivity for Directed Graphs**. In *Proceedings of the IEEE Mediterranean Conference on Control and Automation (MED)*, Platanias-Chania, Crete - Greece, jun. 2013.

[45] Nicolas Chopin and Christian Schafer. **Adaptive Monte Carlo on Multivariate Binary Sampling Spaces**. Working papers, Centre de Recherche en Economie et Statistique, 2010.

[46] Hans Hagen and Georges-Pierre Bonneau. **Variational design of smooth rational Bezier curves**. *Computer Aided Geometric Design*, **8**(5):393 – 399, 1991.

[47] Byungjae Park, Jinwoo Choi, and Wan Kyun Chung. **An efficient mobile robot path planning using hierarchical roadmap representation in indoor environment**. *IEEE International Conference on Robotics and Automation, 2012*, pages 180–186, May 2012.

[48] Anthony Stentz. **Optimal and efficient path planning for partially-known environments**. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 3310–3317. IEEE, 1994.

[49] Carlos E. García, David M. Prett, and Manfred Morari. **Model predictive control: Theory and practice—A survey**. *Automatica*, **25**(3):335 – 348, 1989.

[50] Nancy A Lynch. *Distributed algorithms*. Morgan Kaufmann, 1996.

[51] Lorenzo Sabattini, Valerio Digani, Matteo Lucchi, Cristian Secchi, and Cesare Fantuzzi. **Mission Assignment for Multi-Vehicle Systems in Industrial Environments**. In *Proceedings of the IFAC Symposium on Robot Control (SYROCO). Salvador, Brazil*, August 2015.

[52] Valerio Digani, M.Ani Hsieh, Lorenzo Sabattini, and Cristian Secchi. **A Quadratic Programming Approach for Coordinating Multi-AGV Systems**. In *Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE), 2015. Gothenburg, Sweden*, August 2015.

[53] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2009.

[54] Maren Bennewitz, Wolfram Burgard, and Sebastian Thrun. **Constraint-based optimization of priority schemes for decoupled path planning techniques**. In *KI 2001: Advances in Artificial Intelligence*, pages 78–93. Springer, 2001.

[55] Johan Thunberg, David A Anisi, and Petter Ögren. **A comparative study of task assignment and path planning methods for multi-UGV missions**. In *Optimization and Cooperative Control Strategies*, pages 167–180. Springer, 2009.

[56] Brian P Gerkey and Maja J Matarić. **A formal analysis and taxonomy of task allocation in multi-robot systems**. *The International Journal of Robotics Research*, **23**(9):939–954, 2004.

[57] David W Pentico. **Assignment problems: A golden anniversary survey**. *European Journal of Operational Research*, **176**(2):774–793, 2007.

[58] G Ayorkor Mills-Tettey, Anthony Stentz, and M Bernardine Dias. **The dynamic hungarian algorithm for the assignment problem with changing costs**. 2007.

[59] Dimitra Panagou, Matthew Turpin, and Vijay Kumar. **Decentralized Goal Assignment and Trajectory Generation in Multi-Robot Networks: A Multiple Lyapunov Functions Approach**. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, June 2014.

[60] Matthew Turpin, Nathan Michael, and Vijay Kumar. **CAPT: Concurrent Assignment and Planning of Trajectories for multiple robots**. *The International Journal of Robotics Research*, **33**(1):98–112, 2014.

[61] Edsger W Dijkstra. **A note on two problems in connexion with graphs**. *Numerische mathematik*, **1**(1):269–271, 1959.

[62] V. Digani, L. Sabattini, and C. Secchi. **A Probabilistic Eulerian Traffic Model for the Coordination of Multiple AGVs in Automatic Warehouses**. *Robotics and Automation Letters, IEEE*, **1**(1):26–32, Jan 2016.

[63] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn. **Efficient network-wide model-based predictive control for urban traffic networks**. *Transportation Research Part C: Emerging Technologies*, **24**:122 – 140, 2012.

[64] Dengfeng Sun, Issam S Strub, and Alexandre M Bayen. **Comparison of the performance of four Eulerian network flow models for strategic air traffic management**. *Networks and Heterogeneous Media*, **2**(4):569, 2007.

[65] Alexandre M Bayen, Robin L Raffard, and Claire J Tomlin. **Adjoint-based control of a new Eulerian network model of air traffic flow**. *Control Systems Technology, IEEE Transactions on*, **14**(5):804–818, 2006.

[66] Sandip Roy, Banavar Sridhar, and George C Verghese. **An aggregate dynamic stochastic model for an air traffic system**. In *Proceedings of the 5th Eurocontrol/Federal Aviation Agency Air Traffic Management Research and Development Seminar, Budapest, Hungary*, 2003.

[67] Yan Wan, C. Taylor, S. Roy, C. Wanke, and Yi Zhou. **Dynamic Queuing Network Model for Flow Contingency Management**. *Intelligent Transportation Systems, IEEE Transactions on*, **14**(3):1380–1392, Sept 2013.

[68] D. Miculescu and S. Karaman. **Polling-systems-based control of high-performance provably-safe autonomous intersections**. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 1417–1423, Dec 2014.

[69] Carlos F. Daganzo. **The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory**. *Transportation Research Part B: Methodological*, **28**(4):269 – 287, 1994.

[70] Carlos F. Daganzo. **The cell transmission model, part II: Network traffic**. *Transportation Research Part B: Methodological*, **29**(2):79 – 93, 1995.

[71] Dengfeng Sun and Alexandre M Bayen. **Multicommodity Eulerian-Lagrangian large-capacity cell transmission model for en route traffic**. *Journal of guidance, control, and dynamics*, **31**(3):616–628, 2008.

[72] Ismail Chabini and Shan Lan. **Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks**. *Intelligent Transportation Systems, IEEE Transactions on*, **3**(1):60–74, 2002.

[73] M. Ferrati and L. Pallottino. **A time expanded network based algorithm for safe and efficient distributed multi-agent coordination**. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 2805–2810, Dec 2013.

[74] Reza Olfati-Saber, J. Alex Fax, and Richard M. Murray. **Consensus and Cooperation in Networked Multi-Agent Systems**. *Proceedings of the IEEE*, **95**(1):215–233, January 2007.

[75] Jack Edmonds and Richard M Karp. **Theoretical improvements in algorithmic efficiency for network flow problems**. *Journal of the ACM (JACM)*, **19**(2):248–264, 1972.

[76] F. Oleari, M. Magnani, D. Ronzoni, L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, and C. Fantuzzi. **Improving AGV systems: Integration of advanced sensing and control technologies**. In *Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on*, pages 257–262, Sept 2015.

[77] E. Cardarelli, L. Sabattini, V. Digani, C. Secchi, and C. Fantuzzi. **Interacting with a multi AGV system**. In *Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on*, pages 263–267, Sept 2015.

[78] L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, C. Fantuzzi, and K. Fuerstenberg. **Advanced sensing and control techniques for multi AGV systems in shared industrial environments**. In *Emerging Technologies Factory Automation (ETFA), 2015 IEEE 20th Conference on*, pages 1–7, Sept 2015.

[79] V. Digani, F. Caramaschi, L. Sabattini, C. Secchi, and C. Fantuzzi. **Obstacle avoidance for industrial AGVs**. In *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on*, pages 227–232, Sept 2014.

[80] L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, and C. Fantuzzi. **Multi AGV Systems in Shared Industrial Environments: Advanced Sensing and Control Techniques for Enhanced Safety and Improved Efficiency**. In *Workshop on "Autonomous Industrial Vehicles: From the Laboratory to the Factory Floor", Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[81] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi. **Decentralized coordination enhanced by centralized information: multiple AGVs in industrial application**. In *Workshop "On the centrality of decentralization in multi-robot systems: holy grail or false idol?" Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[82] L. Sabattini, E. Cardarelli, V. Digani, C. Secchi, and C. Fantuzzi. **Multi AGV Systems in Shared Industrial Environments: Advanced Sensing and Control Techniques for Enhanced Safety and Improved Efficiency**. *in "STP1594 - Autonomous Industrial Vehicles: From the Laboratory to the Factory Floor", ASTM*, 2015. in press.