Thesis for the degree of Doctor of Philosophy

# Collision-free path coordination and cycle time optimization of industrial robot cells

Domenico Spensieri

Department of Industrial and Materials Science
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2021

Collision-free path coordination and cycle time optimization of industrial robot cells
Domenico Spensieri
978-91-7905-579-0



Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 5046
ISSN 0346-718X

Department of Industrial and Materials Science
Chalmers University of Technology
SE–412 96 Göteborg, Sweden
Telephone + 46 (0) 31 - 772 1000

Domenico Spensieri is employed by the
Fraunhofer-Chalmers Research Centre for Industrial Mathematics
Chalmers Science Park
SE-41288 Göteborg, Sweden
Telephone + 46 (0) 31 - 772 4252

Cover image: stud welding station and coordination diagrams.

Typeset by the author using LaTeX.

Printed by Chalmers Reproservice
Göteborg, Sweden 2021

*to my parents*

# Abstract

In industry, short ramp-up times, product quality, product customization and high production rates are among the main drivers of technological progress. This is especially true for automotive manufacturers whose market is very competitive, constantly pushing for new solutions. In this industry, many of the processes are carried out by robots: for example, operations such as stud/spot welding, sealing, painting and inspection. Besides higher production rates, the improvement of these processes is important from a sustainability perspective, since an optimized equipment utilization may be achieved, in terms of resources used, including such things as robots, energy, and physical prototyping.

The achievements of such goals may, nowadays, be reached also thanks to virtual methods, which make modeling, simulation and optimization of industrial processes possible. The work in this thesis may be positioned in this area and focuses on virtual product and production development for throughput improvement of robotics processes in the automotive industry. Specifically, the thesis presents methods, algorithms and tools to avoid collisions and minimize cycle time in multi-robot stations. It starts with an overview of the problem, providing insights into the relationship between the volumes shared by the robots' workspaces and more abstract modeling spaces. It then describes a computational method for minimizing cycle time when robot paths are geometrically fixed and only velocity tuning is allowed to avoid collisions.

Additional requirements are considered for running these solutions in industrial setups, specifically the time delays introduced when stopping robots to exchange information with a programmable logic controller (PLC). A post-processing step is suggested, with algorithms taking into account these practical constraints. When no communication at all with the PLC is highly desirable, a method of providing such programs is described to give completely separated robot workspaces. Finally, when this is not possible (in very cluttered environments and with densely distributed tasks, for example), robot routes are modified by changing the order of operations to avoid collisions between robots.

In summary, by requiring fewer iterations between different planning stages, using automatic tools to optimize the process and by reducing physical prototyping, the research presented in this thesis (and the corresponding implementation in software platforms) will improve virtual product and production realization for robotic applications.

**Keywords:** production planning, multi-robot routing and coordination, multi-robot collision avoidance, cycle time optimization.

# Acknowledgments

I would like to thank my academic supervisor Prof. Rikard Söderberg, at the Department of Industrial and Materials Science at Chalmers University of Technology, and my industrial supervisor Dr. Johan Carlson, at the Fraunhofer-Chalmers Centre (FCC), for giving me this opportunity and for inspiring me, both technically and strategically, throughout the entire projects.

I wish to extend my personal thanks to Robert Bohlin for his technical advise on many algorithms and especially on path planning.

I would also like to thank all my co-authors for interesting and fruitful discussions, especially Edvin Åblad, and express my gratitude to the colleagues at FCC, for creating such an inspiring environment.

A special thanks goes to my friends and colleagues Staffan Björkenstam and Tomas Hermansson: Tomas for always encouraging me and helping with the Swedish philosophy; Staffan for the support in our research activities, especially abroad.

Thanks to my family in Sweden and Italy for always and unconditionally supporting me.

This work was carried out at the Fraunhofer-Chalmers Centre, at the Department of Industrial and Materials Science at Chalmers University of Technology and at the Wingquist Laboratory. It is part of the Sustainable Production Initiative and the Production Area Advance at Chalmers University of Technology and was supported by the Swedish Governmental Agency for Innovation Systems. It was also supported by the Smart Assembly 4.0 project and by the Swedish Foundation for Strategic Research (SSF).

Domenico Spensieri

Göteborg, October 2021

# List of Publications

This thesis is based on the following appended papers:

**Paper A.** **Domenico Spensieri**, Edvin Åblad, Jonas Kressin, Johan S. Carlson, Alf Andersson. *Collision-Free Coordination and Visualization Tools for Robust Cycle Time Optimization.* ASME Journal of Computing Science in Engineering, Vol. 21, No. 4, August 2021.

**Paper B.** **Domenico Spensieri**, Robert Bohlin, Johan S. Carlson. *Coordination of robot paths for cycle time minimization.* IEEE International Conference of Automation Science and Engineering, 2013.

**Paper C.** **Domenico Spensieri**, Edvin Åblad, Robert Bohlin, Johan S. Carlson, Rikard Söderberg. *Modeling and optimization of implementation aspects in industrial robot coordination.* Robotics and Computer-Integrated Manufacturing, Vol. 69, June 2021.

**Paper D.** Edvin Åblad, **Domenico Spensieri**, Robert Bohlin, Johan S. Carlson. *Intersection-Free Geometrical Partitioning of Multirobot Stations for Cycle Time Optimization.* IEEE Transactions on Automation Science and Engineering, Vol. 15, Issue 2, pp. 842-851, April 2018.

**Paper E.** **Domenico Spensieri**, Johan S. Carlson, Fredrik Ekstedt, Robert Bohlin. *An iterative approach for collision free routing and scheduling in multirobot stations.* IEEE Transactions on Automation Science and Engineering, Vol. 13, Issue 2, pp. 950-962, April 2016.

Other relevant publications co-authored by Domenico Spensieri:

Edvin Åblad, **Domenico Spensieri**, Robert Bohlin, Ann-Brith Strömberg. *Continuous collision detection of pairs of robot motions under velocity uncertainty.* IEEE Transactions on Robotics, 2021.

**Domenico Spensieri**, Fredrik Ekstedt, Johan Torstensson, Robert Bohlin, Johan S. Carlson. *Throughput maximization by balancing, sequencing and coordinating motions of operations in multi-robot stations.* Proceedings of the 8th NordDesign Conference, Gothenburg, Sweden, 2010.

# Distribution of work

The work on each appended paper was distributed as follows:

**Paper A.** Spensieri outlined the concepts of the method and implemented most of the proposed algorithms (except for outlining, implementing and describing the algorithm in Section 5.1). Spensieri drafted most of the article and reviewed other parts. The other authors contributed with comments and feedback.

**Paper B.** Spensieri outlined the concepts of the algorithm and drafted most of the paper. The other authors contributed ongoing with comments and feedback.

**Paper C.** Spensieri outlined and implemented most of the method and algorithms and drafted most of the paper. Åblad contributed with suggestions about the methods and assisted by testing the algorithms in Section 3.2.1 and giving feedback on the text. The other authors contributed with comments and feedback.

**Paper D.** All authors developed the overall idea. Åblad defined most of the methods in detail, implemented the algorithms and drafted the paper. Spensieri contributed with ideas and input to the method and algorithms and gave comments on the draft of the paper. The other authors contributed with comments and feedback.

**Paper E.** Spensieri outlined and implemented most of the method and algorithms and drafted the paper. The other authors contributed with comments and feedback.

# Contents

# Part I

# Introductory chapters

# Chapter 1

# Introduction

## 1.1 Background and rationale

In the automotive industry, robots are used in a dense and cluttered environment to conduct multiple operations on the Body-in-White (BiW), see (Segeborn, Segerdahl, Carlson, et al. 2010) and (Hömberg et al. 2017). These operations (which we call *tasks* when planning the process) may cover a variety of applications, ranging from welding and sealing to painting and inspection; essentially, assembly and quality assurance processes.

Operations are often defined based on product quality and functional requirements. For example, the location of welding points must fulfill structural requirements, whereas the features to be measured during quality control are chosen based on the key product characteristics. Afterwards, the engineers responsible for the specific process plan robot programs to accomplish all the tasks defined at the previous stage. However, this phase requires a lot of time and high-level competence skills.

Indeed, a Body-in-White (BiW) consists of about 300 steel sheet metal parts and the joining process some 4000 spot-welding points, see (Segeborn, Segerdahl, Carlson, et al. 2010; Segeborn, Segerdahl, Ekstedt, et al. 2014). The workload is distributed among several stations and assigned to hundreds of industrial robots, with robot motions planned to avoid collisions and fulfill cycle time requirements. An example of an assembly line is shown in Figure 1.1, from the Industrial Path Solutions (IPS) simulation software.

These impressive figures justify the need for automatic tools that can be used to support engineers in the various planning phases. Modeling, simulating and optimizing the fundamental aspects of product and production development in a virtual environment (such as a software platform) is one way to achieve that. Moreover, these tools and activities enable a shorter time-to-market for new products and cost savings through avoiding prototyping and testing. Also energy saving can be addressed, see (Wigström et al. 2013).

Digital product realization, as referred to the Wingquist Laboratory, is believed to be the key to efficient and sustainable product and production development, see (Wingquist Laboratory 2017).

The main advantages are:

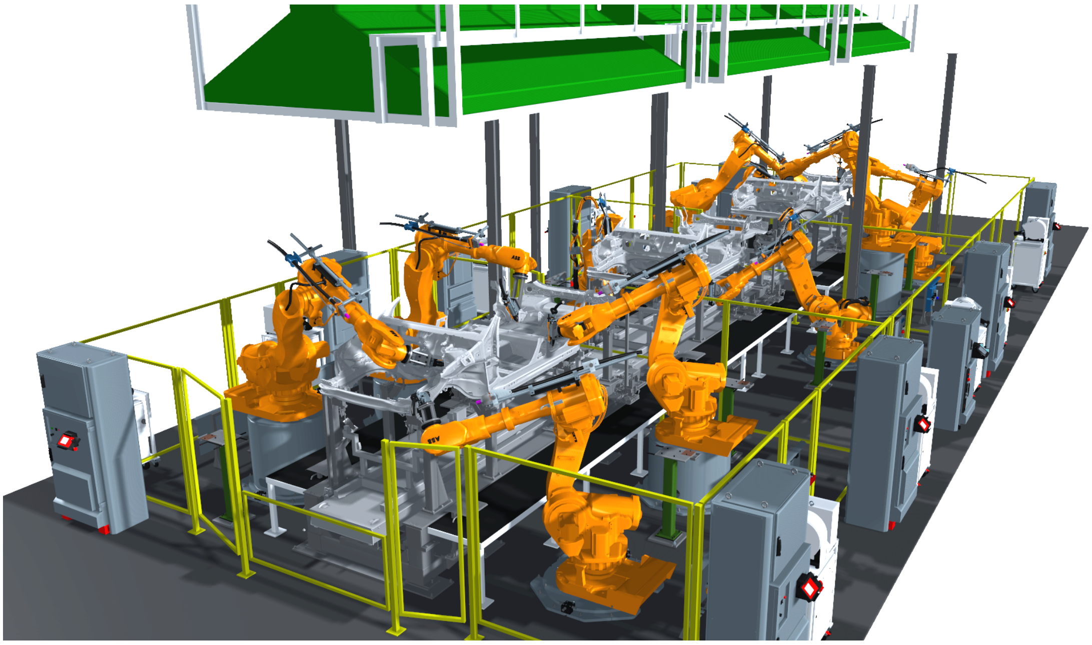


Figure 1.1: Assembly line with three stations and ten robots, modeled in IPS. Courtesy of Volvo Cars Corporation.

- increased product quality,
- decreased commissioning times in production planning,
- increased production rate,
- improved feasibility of the planned operations when first implemented.

Depending on which aspect of the product and production development the tools focus on, they are usually categorized into e.g. *CAD, CAT, CAM, PLM*, and others. Computer-aided design (CAD) software usually aids in the development of a product design, which ranges from mechanical to electronic engineering. Computer-aided tolerancing software (CAT) supports the engineers in analyzing and synthesizing tolerancing issues in complex mechanical assemblies. Computer-aided manufacturing (CAM) software assists the production engineer to control machine tools but is sometimes referred also to more general manufacturing activities. On top of these and other *CAx* tools, there are often Product life-cycle management (PLM) systems managing the entire life-cycle of a product, integrating the process and business levels.

The methods developed in this work are closely related to those which might be used in some of these computer-aided activities. Indeed, most of the research has been integrated (in the form of a demonstrator) with the IPS platform. This software suite offers *CAD* functionalities in terms of cable design and digital manufacturing solutions in the area of robotics, similar to computer-aided engineering (CAE) tools. The specific subject of this thesis is to study and provide models, methods and algorithms for conflict resolution and cycle time optimization in robotics assembly and inspection applications, both considered as parts of the production and engineering process.

The challenges and needs arising in this area were mainly dictated by the organizations supporting this research: namely the Fraunhofer-Chalmers Centre, Chalmers University of Technology and the Wingquist Laboratory. Since many stakeholders have been involved, the next section will provide a brief description of how the research process itself was carried out.

## 1.2 Research approach

As a research student working in industry, it is important to fulfill the stakeholders' various expectations and goals. The scientific community and universities are interested in high-quality research, focusing on such things as scientific contributions in the form of published articles. On the other hand, there might also be applied research institutes who are interested in the applications of such contributions. Moreover, industries may wish to address the optimization of their production processes and go beyond just improving the functions and overall quality of the products/services they sell.

To try and achieve that, the author has striven to adopt a research approach which follows a well-established *research methodology*, namely the design research methodology (DRM), combined with the *strategy* of the center where most of the activities were carried out, the Wingquist Laboratory.

### 1.2.1 Design research methodology

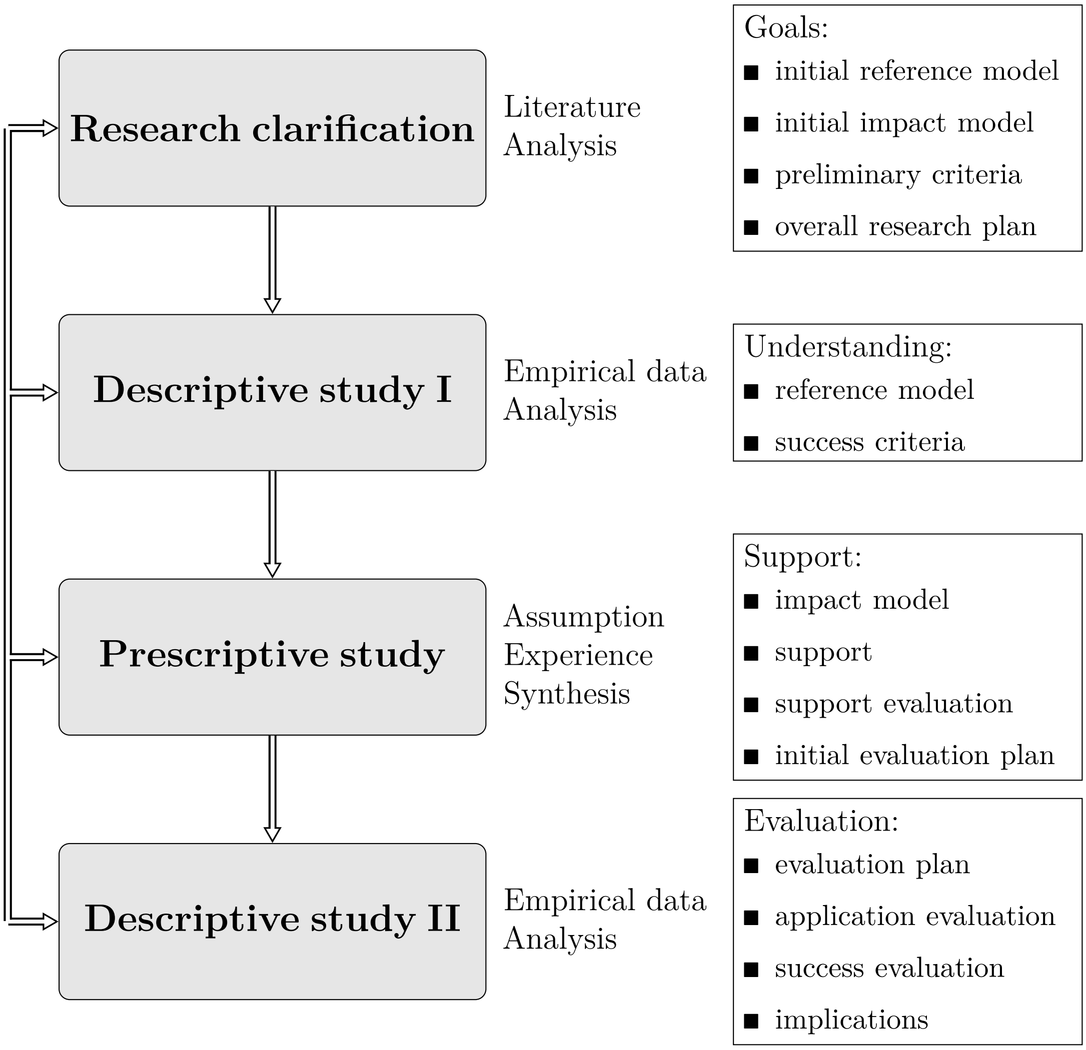
According to (Blessing et al. 2009), design research methodology is a systematic approach consisting of methods and guidelines to support design research. Design research is the development of understanding and support to make design more effective and efficient and thus develop more successful products. It should be noted that this methodology is quite general and dedicated to product design. However, in this thesis it has been adopted as a sound guideline for conducting research about methods and algorithms at high and low levels of detail as needed.

According to DRM, research activities and methods should be divided into four stages, consisting of a criteria formulation and three studies. These stages may be done sequentially or iteratively for successful results and are described below.

**Research clarification:** the existing situation is stated by searching the literature with the desired situation described in the findings/indications supporting the main assumption/idea. An overall research plan is also described. Researchers formulate initial *criteria* that the research is expected to fulfill or as evaluation measures.

**Descriptive study I:** identify the factors that make the description in the previous stage more detailed and that influence the above *criteria*. Researchers analyze empirical data and evaluate their understanding of the current situation.

**Prescriptive study:** develop an impact theory or model as basis for systematic development of methods, usually aiming for a proof-of-concept.




**Descriptive study II:** identify whether the methods and tools may be used in the intended situations (*application evaluation*) and whether/how they fulfill the success criteria (*success evaluation*).

#### Validation and verification

An important aspect within *DRM* is the evaluation phase. This appears at each stage, being applied to the support and the application, and consists in determining whether the success criteria have been fulfilled. The evaluation may consist of several activities and, among them, particular care is reserved to validation and verification. Note that these two terms are often used interchangeably so at this point a clarification is made. Since this work may be classed as method and algorithm development, the definitions of validation and verification processes are very close to those used in the field of computer science. According to (Boehm 1979) and as also stated in (Blessing et al. 2009, Chapter 4):

- *validation* is ensuring that one has built the correct things and answered the question, “was the right system built?”;

- *verification* is ensuring that one has built the thing correctly, answering the question, “was the system built right?”.

Validation involves checking that the methods and implemented algorithms meet the “customer/stakeholder” requirements. For example, the Institute of Electrical and Electronics Engineers (IEEE) defines validation as the “process of providing supporting evidence that the software satisfies system requirements allocated to software and solves the right problem”, see (“IEEE Standard for Software Verification and Validation” 1998).

Verification, on the other hand, involves checking that the methods and the algorithms implemented conform to the initial specifications. For example, the same IEEE defines as the “process providing supporting evidence that the software and its associated products comply with requirements {...}, satisfy standards {...}, and establish a basis for assessing the completion of each life cycle activity”, see (“IEEE Standard for Software Verification and Validation” 1998).

Methods may also be verified “by acceptance”, for example by being described in articles accepted in peer-reviewed conferences or journals.

### 1.2.2 Wingquist Laboratory research strategy

The work for this thesis was conducted at the Wingquist Laboratory (WQL), see (Wingquist Laboratory 2017), and its VINNOVA excellence center: Wingquist Laboratory VINN Excellence Centre, see (Wingquist Laboratory VINN Excellence Centre 2017). The Centre focuses on virtual product realization, with research topics formulated based on a scientific challenge and an identified industrial need. This involves synthesizing the intended research focus and normally involves formulation of *research questions*, see Section 1.3.1.

In addition to traditional scientific results like academic publications, the research projects also result in a demonstrator (a suggested functionality or working procedure) developed by the research team. Figure 1.2 illustrates the different components of a successful project within the Wingquist Laboratory.

Given these background premises, this author will now narrow the scope of the research and present this thesis’ detailed areas of investigation and contribution, from both the scientific and industrial points of view.

## 1.3 Research scope

One of the main challenges in production planning is satisfying a given production rate to ensure that business goals are achieved. A key aspect is guaranteeing that a given cycle time is achieved in the assembly process. As initially stated in Section 1.1, assigning the tasks to the right robots, sequencing them avoiding collisions, all whilst minimizing cycle time is quite a challenge when done manually. Thus, automatic or semi-automatic tools are highly relevant.

This general problem has become very attractive among researchers, due to its enormous impact on production rates. Nowadays, it is seeing a phase in which a lot

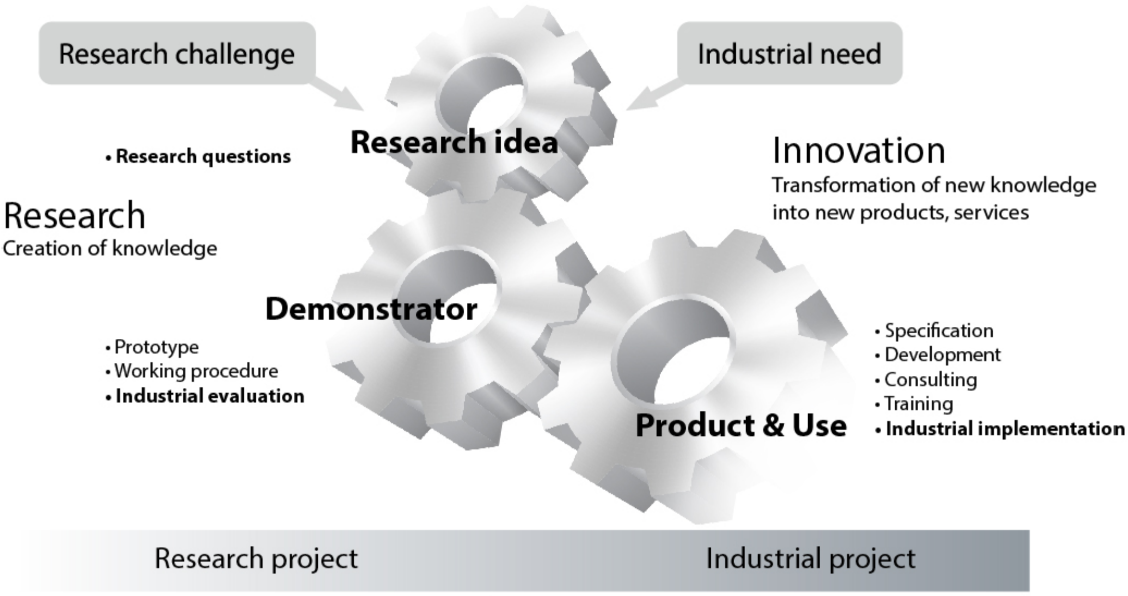



Figure 1.2: Wingquist Laboratory research strategy.

of progress is being made, see (Pellegrinelli et al. 2017; Cantos Lopes et al. 2017; Skutella et al. 2011; Rambau et al. 2014; Landry et al. 2013; Xin et al. 2020). One of the most difficult features to deal with is the interconnection between geometric and time aspects, which makes the problem hard to solve. Thus, the entire problem is often decoupled into smaller ones, by trading completeness with speed. This means that high-quality solutions to major problems to large instances may be obtained within reasonable time. A typical decoupling strategy, see (Segeborn, Segerdahl, Carlson, et al. 2010), consists of the following main steps.

**Task planning:** finding multiple robot configurations able to perform each individual task of the process.

**Load balancing:** distributing tasks among the robots in a balanced way to minimize cycle time.

**Tasks sequencing:** finding task sequences for each robot, to minimize cycle time.

**Path planning:** creating robot motions between pairs of configurations so that they do not collide with the static environment.

**Path coordination:** creating robot coordination schemes which prevent moving robots from colliding with each other and which minimize cycle time.

**Program generation:** creating robot programs to be executed by robot controllers and interacting with a PLC, to minimize the number of communication delays.

Each step has its own challenges, for which reason it has been difficult, during the years of these doctoral studies, to balance in-depth investigation of specific aspects with overall approaches, still tackling all the steps at a certain level of detail. From a scientific point of view, the challenges center on integrating and balancing combinatorial optimization, computational geometry, robotics and graph-searching with modeling that considers industrial requirements and expectations. In the spirit of the Wingquist Laboratory, three main research questions were considered, to find a good compromise between all the research directions which might have been followed. These questions are formulated and described in next section.

### 1.3.1 Research questions

**Research question 1**

How might the equipment utilization of multi-robot stations be improved?

This question is quite general and there are several ways to approach it. For example, one might investigate whether it is possible to complete the process by eliminating one robot, or by avoiding the PLC. Other ways to tackle the problem include minimizing energy or cycle time and using the relevant equipment to maximize throughput. This research topic is investigated in *all* appended papers but especially in Paper B, where it is tackled directly and in Papers D and E, where planning follows a more general framework.

However, in many industrial applications, robots are constrained to move along fixed geometric paths. This is due to restrictions in the processes they carry out, such as sealing or painting applications. Sometimes, even point-to-point motions might be very restricted, due to low clearance with the surroundings, thus limiting the freedom of motion. These observations give rise to the second research question.

**Research question 2**

How might robots avoid mutual collisions when executing geometrically predefined robot programs, so as to minimize cycle time?

This problem is limited to robot paths fixed within the configuration space, the trajectories of which (their paths in time) are not yet defined. The goal is to introduce waiting times or slow down robot motions (velocity tuning) avoiding mutual collisions and optimize cycle time. In addition to the above factors, many other applications encounter similar problems, such automated guided vehicles (AGVs) on fixed tracks (Olmi, Secchi, and Fantuzzi 2008; Olmi and Secchi 2011) or air traffic management (ATM), see (Pallottino et al. 2002). Moreover, the assumption of fixed paths is often used in solving more complex problems that are prohibitive from a computational point of view. In these approaches, the problem is decoupled into simpler sub-problems, which are then solved separately and coupled together according to a given strategy. Due to its potential impact on the scientific community and relevance to industrial robot cells, this is the main research question of this thesis. It is mainly investigated in Paper B, with a complete algorithm and partially in Papers A and C. In Figure 1.3 a typical stud welding multi-robot station in the automotive industry, with two robots colliding along their paths.

At any rate, more practical aspects must be considered when implementing solutions to this problem. One of the most relevant is the time delay introduced when stopping a robot's motion to ask a PLC for permission to continue executing its program. This aspect is considered in the next research question.

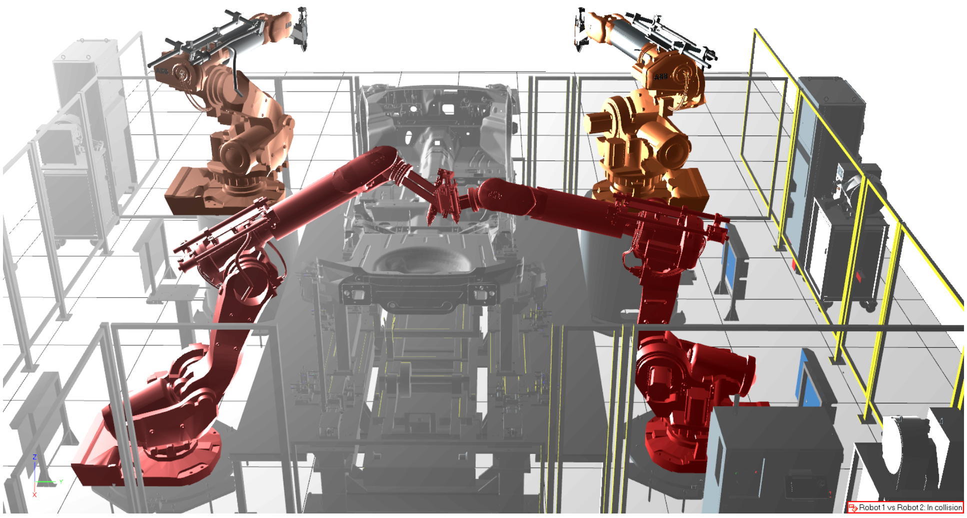


Figure 1.3: Collision between two robots along their defined paths.

**Research question 3**

How might relevant industrial implementation requirements be modeled to optimize cycle time?

The time delays introduced when implementing an optimal solution in industrial scenarios are optimized in a post-processing step. This is an attempt to minimize the number of points along a robot path at which synchronization with other robots is needed. This aspect is mainly investigated in Papers A and C. A more drastic approach is to completely eliminate the need for robot synchronization by planning robot motions that never intersect with each other and independently of their velocity. This implies that the robots' swept volumes during their program execution are disjointed. A solution to such a problem is proposed in Paper D.

### 1.3.2 Success criteria

From industrial users perspective, expectations are mainly ones of decreasing commissioning time and obtaining high-quality solutions to their problems. This last aspect is highly relevant from a scientific point of view. Indeed, one might argue that fast algorithms which solve a given problem also aid the first goal: faster robot planning synthesis. Based on these observations, four main success criteria have been formulated and are described below.

**High-quality solutions:** the methods developed should be able to compute high-quality solutions for industrially relevant problems, bearing in mind the complexity of the CAD models and processes involved (no. of tasks, no. of robots, and so on). By “high-quality” it is meant optimal or close to it (when this is known), or feasible in case the problem is so complex that straightforward comparisons are not easy to obtain.

**Solving speed:** the methods developed should provide solutions fast, for use in iterative planning and management activities and in iterative algorithms solving larger problems. By “fast” it is meant in order of minutes/hours for complex methods involving industrial scenarios, and in order of seconds or within computing times comparable to existing packages able to solve the same problem.

**Decrease commissioning time:** the methods developed should provide fast solutions and should be available in software demonstrators (as proof of concept) to be used, in a relatively easy way, by process engineers. They should be able to address industrial scenarios and should not require advanced mathematical knowledge.

**Scientific contribution:** the methods developed should be highly relevant scientifically such that they may contribute to the research community and be shared in international peer-reviewed publications.

## 1.4 Outline

This introduction, **Chapter 1**, demonstrated the rationale, background and scope of the work. **Chapter 2** briefly introduces the *frame of reference* for the thesis: it gives a short description of the underlying problems and algorithms and identifies the research/industrial gap leading to the main research questions. In **Chapter 3** the *contributions* of this thesis are summarized: each appended article is briefly described and put into the context of the general research problem. **Chapter 4** highlights the main scientific and industrial *results*, discussing the connections with the initial research questions. **Chapter 5** concludes this thesis by recapping the main findings in a general perspective and suggesting possible directions for *future work*.

# Chapter 2

# Frame of reference

This chapter gives a brief overview of models and algorithms used throughout this work. This serves as a frame of reference to position the thesis in a more general research context.

## 2.1 Single agent sequencing

In robotic assembly cells, the order in which operations are carried out may influence both product quality and production rate. Indeed, sheet metals spot welding introduces geometric variation and different welding orders may produce different results throughout the entire assembly (Carlson et al. 2014). Moreover, the total cycle time is directly influenced by the motion time from one process location to another and, consequently, by the order in which they are visited. This applies not only to assembly tasks but also to painting, sealing and inspection processes.

In considering cycle time for a single robot, the problem may often be modeled as a traveling salesman problem (TSP).

### 2.1.1 Traveling salesman problem

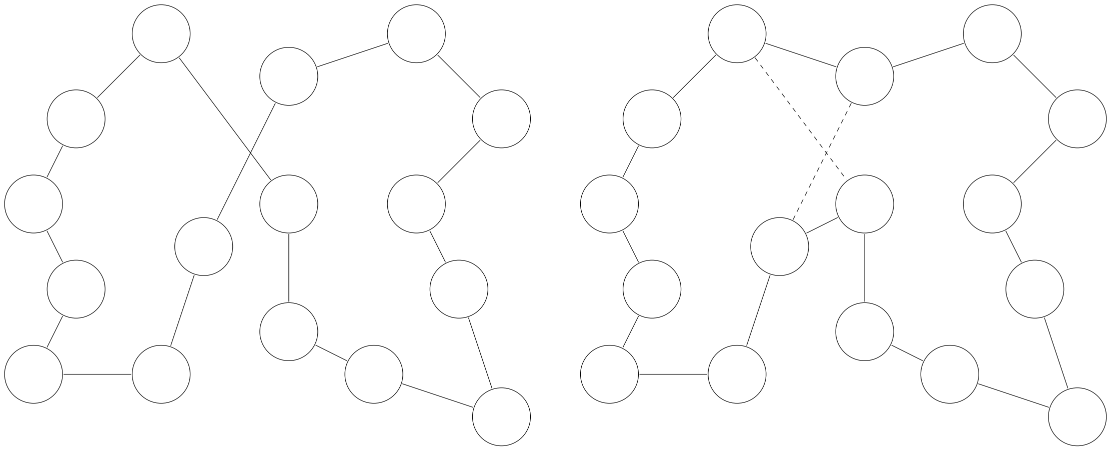
As is well known, this is one of the most intensively studied problems in combinatorics: given a set of cities $\mathcal{G} = \{1, 2, \ldots, n\}$ and distances $c_{ij}$ between cities $i$ and $j$, the problem is to find the shortest tour for a salesperson, starting from city 1, visiting each city in $\mathcal{G}_1 = \mathcal{G} \setminus \{1\}$ once and returning to the start city.

The decision version of this problem belongs in the class of NP-complete problems, see (Garey et al. 1990). An easily implemented algorithm for small (up to about 20) instances is the dynamic programming approach by Held and Karp, (Held et al. 1961), where the equations used for the recursion are:

$$\gamma(\{j\}, j) = c_{1j}, \tag{2.1a}$$

$$\gamma(\mathcal{S}, j) = \min_{i \in \mathcal{S}_j} \{\gamma(\mathcal{S}_j, i) + c_{ij}\}. \tag{2.1b}$$

Here, $i, j \in \mathcal{G}_1$, $\mathcal{S} \subseteq \mathcal{G}_1$, $\mathcal{S}_j = \mathcal{S} \setminus \{j\}$ and $\gamma(\mathcal{S}, j)$ denotes the accumulated distance, starting at city 1, visiting all cities in $\mathcal{S}_j$ and ending at $j \in \mathcal{S}$.



(a) Original tour. (b) Tour after a 2-opt move.

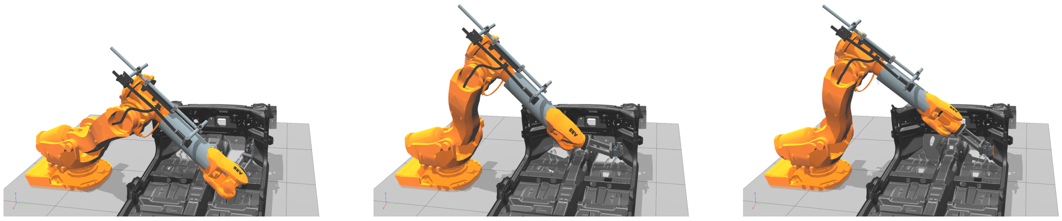
Figure 2.1: 2-opt moves are used as local search in TSP solvers.

For larger instances, exact algorithms may be based on minimum spanning tree (MST) relaxations combined with sub-gradient ascent methods and branch and bound, see (Held et al. 1971). The most successful methods are based on integer linear programming (ILP) formulations solved by tailored branch and cut algorithms, see Concorde (Applegate et al. 2019).

However, one may obtain high-quality solutions for many huge instances, without proof of optimality, by using heuristic algorithms based on local search acting on graphs in which cities (or process locations for a robot) are represented by nodes and edges between nodes represent paths between cities (or robot configurations), weighted according to their distance (or robot motion times). The most successful heuristics are based on $k$-opt moves in which an initial tour is successively modified by replacing $k$ edges, see (S. Lin et al. 1973; Helsgaun 2006). An example of 2-opt move is illustrated in Figure 2.1. These solutions are also very useful as starting tours in exact approaches, for speeding up computations.

There are several variants of the TSP (for an overview see the book (Gutin et al. 2002)), which add some constraints, slightly modifying the definition. For example, precedence constraints between cities might be introduced, transforming the TSP into the well known sequential ordering problem (SOP), see (Escudero 1988). This variant may be used do model robot welding processes, in which some points must be welded before others to guarantee a certain geometric quality in the final assembly. These precedence constraints can also be used to model priorities on points in robotic inspection applications. These points are allowed to be measured only after others have been, to build a local reference systems on-the-fly, see coordinate measuring machines (CMM) applications described in (Salman et al. 2016).

One TSP variation that often occurs is the generalized traveling salesman problem (GTSP), in which cities are clustered into groups with the aim of finding the minimum cost tour whilst visiting exactly one node from each group, see (Laporte et al. 1987).



(a) Configuration one. (b) Configuration two. (c) Configuration three.

Figure 2.2: Three different configurations to perform a stud welding task by the same robot.

### 2.1.2 Generalized TSP

It turns out that the GTSP model fits robot sequencing problems perfectly. Each task can be carried out by one robot in many ways. Indeed, several robot configurations may be found by using the redundancy of typical six degrees of freedom (dofs) industrial manipulators and the dofs associated to the task to be processed. For example, in many applications it is important that the robot tool center point (TCP) matches the position part of a task (defined by a frame), whilst still able to rotate around a frame's axis. In Figure 2.2 three different robot configurations are shown, used to perform one single stud welding task.

The optimal sequence of tasks and the optimal cycle time are given by solving the corresponding GTSP. As for the original TSP, a dynamic programming approach may also be adopted for this variant. Indeed, in order to consider several cities within each group, equations (2.1) may be generalized (see (Carlson et al. 2014)). They are modified such that $\mathcal{G}$ is now the set of groups and the computation of the minimum is extended to consider each node in a group. The recursions are then computed by repeating the procedure for each node in the start group.

However, due to memory requirements, only small instances may be solved (up to about 20). For exact solutions to larger instances, a very powerful algorithm is the one devised by (Fischetti et al. 1997), in which Lagrangian relaxation is used in combination with effective cuts in an ILP formulation.

Regarding heuristic algorithms working directly on the GTSP formulation, the ones that generalize the Lin-Kernighan local search are highly efficient, see (Karapetyan et al. 2011).

Moreover, there are a number of exact transformations of GTSP into asymmetric TSP, see (Noon et al. 1993) for the most-used one. These transformations may be used either to apply exact TSP algorithms to them, like the ones mentioned in the previous Section 2.1.1, or to exploit slightly modified efficient TSP heuristics, see (Helsgaun 2015).

In this thesis, GTSP algorithms are used to solve many sequencing problems, see Papers D and E, to optimality or by exploiting efficient heuristics, see (Ekstedt et al. 2009), similar to (Karapetyan et al. 2011).

Nevertheless, occasionally there will be a robot that does not match cycle time requirements. Or it may not be possible to place the robot so that all tasks within that robot's workspace may be reached. These cases require multiple robots. Such multi-robot stations can be optimized by using multiple TSP (MTSP) models.

## 2.2 Multiple agents sequencing

In an MTSP, each city should be visited once by one of the salespeople, see (Toth et al. 2002). A more common name for this problem is vehicle routing problem (VRP), see (Toth et al. 2002), in which the goal is to find an optimal set of routes for a fleet of vehicles (robots, in our case) so that they may deliver to a set of customers (tasks, in our case). Thus, if only one vehicle is present, the problem reduces to a standard TSP.

Usually, the objective function that has to be minimized is the sum of all distances traveled (robot motion times). However, to adapt it to our robotic assembly stations and production rate requirements, the goal needs to be changed to *minimizing the maximum* distance driven by each vehicle. Moreover, each robot has various alternative ways of carrying out a task. This introduces another level of complexity, giving rise to what is sometimes referred to as *generalized* VRP (GVRP), see (Ghiani et al. 2000).

Another necessary generalization is that the distance between tasks actually depends on which robot is assigned them. This leads to different distance matrices for each robot. The resulting VRP problem is sometimes referred to as *heterogeneous*, see (Baldacci et al. 2008; Riazi et al. 2013). To summarize, if there were to be a concise way of referring to the entire problem, then the term *min-max heterogeneous GVRP* might be used.[1]

The problem is computationally demanding and has been tackled in this thesis by decomposing it into several sub-problems. One of the decisions to be made is assigning a given task/operation to a specific robot. This assignment relies heavily on the task feasibility of each robot and has major impact on the makespan. This is because the time to perform a specific process within a given task is often greater than the robot motion time spent from one task to another. This is another reason why such a decomposition has been chosen. Indeed, the property may be exploited to solve the overall problem. For example, this might be done by choosing a branch and bound (B&B) method working on the assignment variables, or by iterating between assigning tasks to robots and sequencing them.

Thus, one might formulate a relevant problem by neglecting the motion times between tasks and considering only the task processing time. In this case, given a set of $N_R$ robots $\mathcal{R} = \{1, \dots, N_R\}$, the goal is to minimize the total cycle time $c$, which is the maximum of all robots' total processing times $c_r$, *i.e.* minimize $\max_{r \in \mathcal{R}}\{c_r\}$.

We call this problem *load balancing*.

[1]Please note, this thesis does not deal with the direct solution to such problems in its complete version. However, the problem is introduced to better understand the core rationale for the algorithms proposed in the appended papers.

### 2.2.1 Load balancing

Given a set of $N_T$ tasks $\mathcal{T} = \{1, \ldots, N_T\}$, a mathematical formulation for this problem may be created using a mixed integer linear programming (MILP) model as follows:

$$
\begin{aligned}
\text{minimize} \quad & c && && \text{(2.2a)}\\
\text{s.t.} \quad & \sum_{i \in \mathcal{T}} c_i^r x_i^r \leq c, && r \in \mathcal{R} && \text{(2.2b)}\\
& \sum_{r \in \mathcal{R}} x_i^r = 1, && i \in \mathcal{T} && \text{(2.2c)}\\
& x_i^r \in \{0, 1\}, && i \in \mathcal{T}, r \in \mathcal{R}. && \text{(2.2d)}
\end{aligned}
$$

Here, $c_i^r$ is the time it takes for robot $r$ to perform task $i$, $c$ is the total cycle time (or makespan) and the decision variables $x_i^r$ take the value 1 if robot $r$ performs task $i$, 0 otherwise. It is a well known problem and appears with the name unrelated parallel machine problem (UPMP) in the operations research literature and $R||C_{\max}$ in the scheduling community, see (Åblad, Strömberg, et al. 2021).

Problem (2.2) is particularly hard to solve due to the min-max objective function. One way to understand the reasons behind it is to compare the Lagrangian relaxation of constraints (2.2b) vs. the relaxation of constraints (2.2c). In the first case, one obtains a directly solvable problem: for each task $i \in \mathcal{T}$, let $x_i^r = 1$, where $r \in \arg\min_{r \in \mathcal{R}} \lambda^r c_i^r$ and $\lambda^r$ is the multiplier relative to the makespan constraint for robot $r$. In the second case (when keeping the makespan constraints (2.2b)), a problem is obtained that, for a fixed value of $c$, may be decomposed into $N_R$ binary knapsack problems. The binary knapsack problem is NP-hard, even if, in practice, its solution may be efficiently obtained by dynamic programming or B&B algorithms. For the exact solutions of this problem, see (Åblad, Strömberg, et al. 2021).

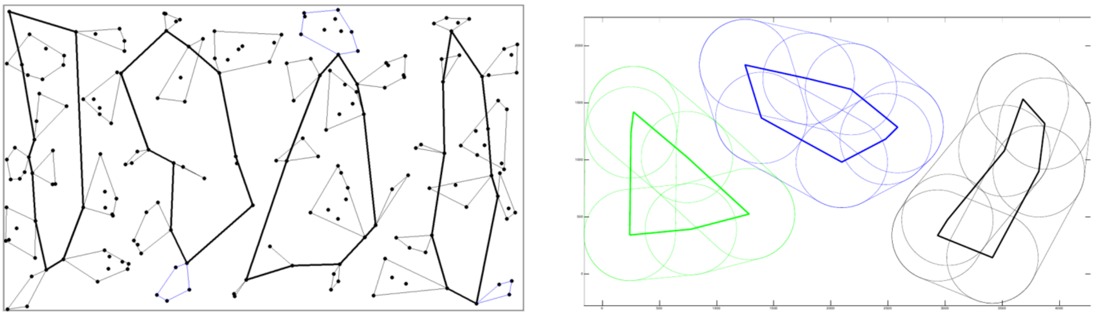
This problem is not the main focus of this thesis but plays an important role in Papers D and E.

### 2.2.2 Conflicts between agents' paths

These models do not capture the fact that there might be conflicts (usually geometric collisions) between robots moving from one task to another. This additional property needs to be modeled and drastically complicates the problem.

When there are collisions, the problem may be tackled from a completely different perspective, regarding it as a multidimensional path-planning problem in which the dofs for each robot are added together; see (Isto et al. 2006) for PRM in high-dimensional spaces. However, in the applications studied in this thesis, robots often have a low clearance to the environment and there is a high risk that current direct algorithms are unable to find good enough solutions in a reasonable time, see (LaValle 2006, Ch. 7).

Therefore, decoupled approaches are often used. Three of the most suitable methods for avoiding the robots being in the same area at the same time are:



(a) Graph representation of an MGTSP.

(b) Graph representation of an MGTSP with some conflicting edges.

Figure 2.3: MGTSP without and with conflicts.

**re-routing** by re-ordering the tasks, while keeping the geometric path from task $i$ to task $j$ unmodified;

**path re-planning** consists in changing the routes of the robots by modifying the geometric path from task $i$ to task $j$;

**path coordination** by introducing waiting times along the routes, to avoid that the robots occupy the same space at the same time, while keeping the geometric path from task $i$ to task $j$ unmodified.

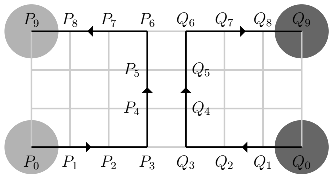
When the first method is used and if extra simplifications are made, then the problem may be approximated as a matter of single-agent sequencing. Indeed, in Paper E the problem is transformed into an artificial GTSP, the solution to which provides a sub-optimal cycle time for a collision-free multi-robot system.

The second method is subject to several variations, see (LaValle 2006, Ch. 7). For example, a prioritized strategy may be used, where each robot is given a priority, with lower priority robot paths planned which treat the previously planned paths as obstacles. Another interesting method is that proposed in (Saha et al. 2006), which combines the previous two methods. It plans in a combined space consisting of the robot configuration space with an additional dof, tuning the velocity of the previously planned paths.

The third approach is the major topic of this thesis and may be incorporated in iterative algorithms. Essentially, it exploits the time aspect to make colliding robot paths into collision-free ones, see also (LaValle 2006). In this context, the problem of introducing waiting time into predefined paths (or tuning the velocity) to avoid collisions will be referred to as path coordination problem (PCP). The method adopted in this thesis is sometimes referred to as fixed-path coordination, to distinguish it from more powerful methods (such as roadmap coordination, which requires multiple paths to act upon).

## 2.3 Path coordination

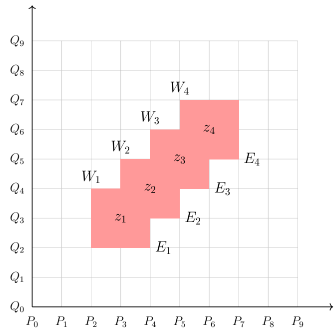
In the PCP, the analysis is restricted solely to those dofs available to avoid collisions; specifically, tuning the velocity of the paths. Each robot is constrained to follow a




(a) Simple case for 2 AGVs.

| *ROBOT 1* | *ROBOT 2* |
|---|---|
| MoveTo $P_1$ | MoveTo $Q_1$ |
| MoveTo $P_2$ | MoveTo $Q_2$ |
| MoveTo $P_3$ | MoveTo $Q_3$ |
| MoveTo $P_4$ | MoveTo $Q_4$ |
| MoveTo $P_5$ | MoveTo $Q_5$ |
| MoveTo $P_6$ | MoveTo $Q_6$ |
| MoveTo $P_7$ | MoveTo $Q_7$ |
| MoveTo $P_8$ | MoveTo $Q_8$ |
| MoveTo $P_9$ | MoveTo $Q_9$ |

(b) Robot programs.




(c) Coordination diagram.

Figure 2.4: Case with two robots and four overlapping square collision zones.

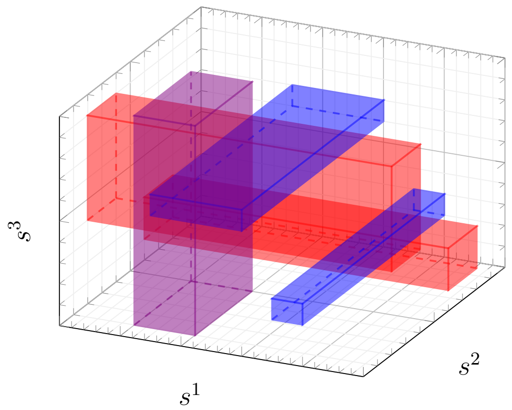
path, modeled by a continuous mapping $\boldsymbol{\tau}(s) : S \to C_{\text{free}}$, where $S = [0, T]$ and $T$ corresponds to the path execution time (when the path is executed at its maximum speed) and $C_{\text{free}}$ is robot's configuration space being collision-free with the static environment $E$, usually a six-dimensional space for typical industrial manipulators. This means that $A(\boldsymbol{\tau}(s)) \cap E = \emptyset, \forall s \in S$, where $A$ defines the volume occupied by the robot at configuration $\boldsymbol{\tau}(s)$.

The *Path Coordination Space* (PCS) for $N_{\text{R}}$ robots is defined as $\overline{S} = S^1 \times S^2 \times \ldots \times S^{N_{\text{R}}}$ (the superscript is relative to the robot index). In this space, the collision-free region is defined as $\overline{S}_{\text{free}} := \{\boldsymbol{s} = (s^1, s^2, \ldots, s^{N_{\text{R}}}) \in \overline{S} \mid A^i(\boldsymbol{\tau}^i(s^i)) \cap A^j(\boldsymbol{\tau}^j(s^j)) = \emptyset, \forall i \neq j\}$ and the obstacle region $\overline{S}_{\text{obs}} := \overline{S} \setminus \overline{S}_{\text{free}}$. If a minimum distance $\underline{\delta}$ between robots is required, then $\overline{S}_{\text{free}}(\underline{\delta}) := \{\boldsymbol{s} = (s^1, s^2, \ldots, s^{N_{\text{R}}}) \in \overline{S} \mid \text{dist}(A^i(\boldsymbol{\tau}^i(s^i)), A^j(\boldsymbol{\tau}^j(s^j))) \geq \underline{\delta}, \forall i \neq j\}$, where $\text{dist}(A^i, A^j)$ is the Euclidean distance between robots $i$ and $j$. The PCP consists of finding a monotone continuous path $\Phi(t) : [0, 1] \to \overline{S}_{\text{free}}$, from $\Phi(0) = (0, 0, \ldots, 0)$ to $\Phi(1) = (T^1, T^2, \ldots, T^{N_{\text{R}}})$. The monotonic property models the constraint that, during the robot programs, instructions are not executed backwards.

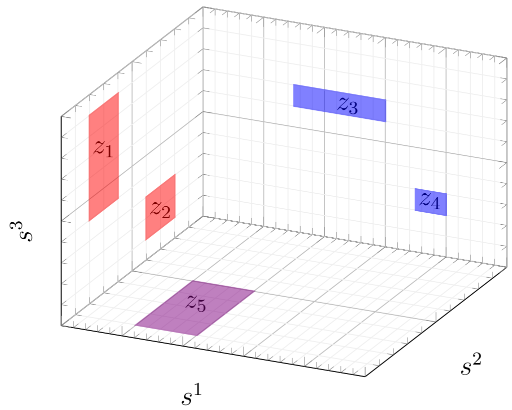
The coordination diagram (CD), introduced in (O'Donnell et al. 1989), is used to model the PCS when the problem involves only two robots. Each axis in the CD is usually discretized, based on the possibility of a robot $r$ stopping at those discrete points $V^r = 1, \ldots, N^r$ along its path, usually named via-points. These model the property that motion instructions cannot be interrupted whilst being executed. A practical way to control the velocity and scheduling is to introduce synchronization instructions before and/or after motion instructions.

A *collision zone* in a CD for robots $i$ and $j$ is defined as a rectangular area $z = (w^i, e^i) \times (e^j, w^j) \subset S^i \times S^j$ within which a collision occurs, defined by the "west/east" corners $W(w^i, w^j)$ and $E(e^i, e^j)$. An example appears in Figure 2.4, with two circular shaped robots shown moving along their paths, from $P_0$ to $P_9$ respectively and from $Q_0$ to $Q_9$, according to their programs in Figure 2.4b. The CD showing where collisions occur (red areas) appears in Figure 2.4c.

A CD in several dimensions is called Generalized Coordination Diagram (GCD). The fact that only two robots are sufficient to define collisions gives a "cylindrical" structure (parallelepipeds if the path are discretized) to the *GCD*, see (LaValle and




(a) GCD visualization for three robot paths.




(b) Collision zones in the pairwise CDs.

Figure 2.5: Rectangular parallelepiped showing the “cylindrical” structure of collision area in the Generalized Coordination Diagram.

Hutchinson 1998) and (LaValle 2006, Ch. 7). This may be seen in Figure 2.5a, where three paths for robots $\mathcal{R} = \{1, 2, 3\}$ collide in the collision zones shown in Figure 2.5b.

### 2.3.1 Practical issues

In practice, optimal coordination is implemented by introducing safeguards in the robot programs. Indeed, relying solely upon velocity tuning the paths does not account for uncertainties in the timing of robot motions and may lead to collisions. Essentially, these instructions allow one robot to wait until another has passed a predefined location. In theory, this exchange of information might only take place between robots. However, in industrial practice each robot asks a PLC for permission to continue its program and, likewise, communicates to the PLC that it has reached a certain location, or program instruction. Thus, time delays are introduced:

- delays due to the time needed for a robot to stop (and await a response from a PLC);
- delays due to the communication time with the PLC.

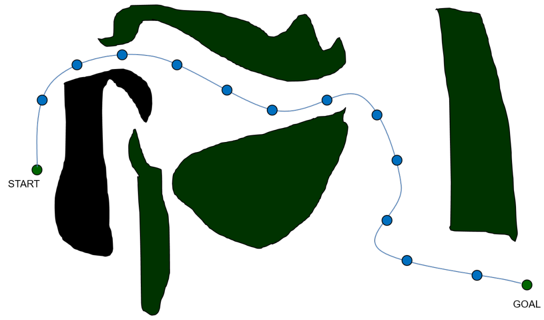
This aspect introduces relevant cycle time increases and is considered in Papers A, C and D.

The next chapter provides a brief introduction to path planning, focusing on those uses which are related to this thesis.

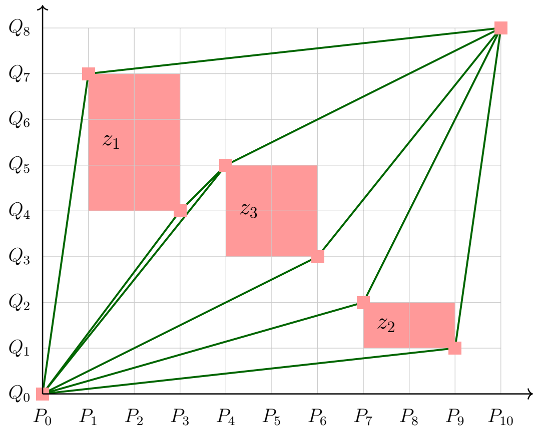
## 2.4 Path planning and collision-free verification

### 2.4.1 Path planning

In its pure form, the path planning problem may be defined as finding a continuous path for an agent (robot, rigid body and so on), from a starting configuration




(a) Smooth path from start to goal in configuration space with obstacles.




(b) Two dimensional visibility graph for path coordination of two robots.

Figure 2.6: Collision-free path in configuration space and visibility graph for the path coordination space.

to a target one, whilst avoiding collisions with the environment, see Figure 2.6a. Although the agent workspace is the 3D world (usually called the task/operational space to differentiate it clearly from the configuration space, see below), planning of collision-free motions usually takes place in the configuration space, which may be informally defined as the set of transformations that a robot with $n$ dofs may assume. For a typical industrial robot, consisting of a serial kinematic chain of six revolute joints with a limited interval of motion, the configuration space $\mathcal{C}$ is the product of all single-joint spaces and may be represented by $\mathcal{C} = \mathbb{R}^6$.

Even for a polyhedral agent, the problem belongs to PSPACE, as shown by (Canny 1988). Thus, in practice, approximations are used, with heuristic algorithms trading off completeness for practical efficiency. One of the most-used and successful techniques is based on sampling the configuration space and searching for a path that connects the samples in a collision-free manner, see (LaValle 2006; Karaman et al. 2011) for an overview. The main algorithms adopted in this class are the probabilistic roadmap method (PRM), see (Kavraki et al. 1996) and the rapidly-exploring random trees (RRT), see (LaValle and Kuffner 2000). These two approaches differ in the way sampling is done and how they construct a graph.

Once a graph is built, it becomes possible to search for optimal paths, according to a given measure, by using classical search algorithms. The most influential one in the field of motion planning is definitely the A*, see (Nilsson 1980). This is an evolution of the Dijkstra algorithm, where an estimate of the goal configuration cost is added to guide the search towards a promising area of the configuration space. For example, in this thesis, A* is used to search the visibility graph that is built to model the path coordination problem for two robots, as described in Paper B; see Figure 2.6b.

In general, path planning is a component of most of the methods proposed in this thesis, however it has not been a subject for scientific contributions. It is used as a tool to find motions between process configurations in the presence of obstacles and to estimate robot motion times.

To plan collision-free motions, the environment needs to be mapped onto the agent configuration space as forbidden areas. This mapping may be done by placing the agent in several configurations and checking whether or not a collision occurs. The problem of collision detection in 3D is a wide research area in computational geometry, see (M. C. Lin et al. 1996). Thus, a brief overview is provided here, focusing on aspects closer to the subject of the thesis.

### 2.4.2 Collision checking multi-robot paths

Objects in virtual environments are often represented by primitives like triangles, polygons, NURBS and so on. Objects may contain millions of triangles, thus making it impractical to check pairwise primitives for collisions. Moreover, besides collision detection, it is very important to compute exact and approximate distances. A powerful approach to these computations uses hierarchical bounding volumes (HBV) to model objects, with various techniques having been devised to traverse these hierarchies.

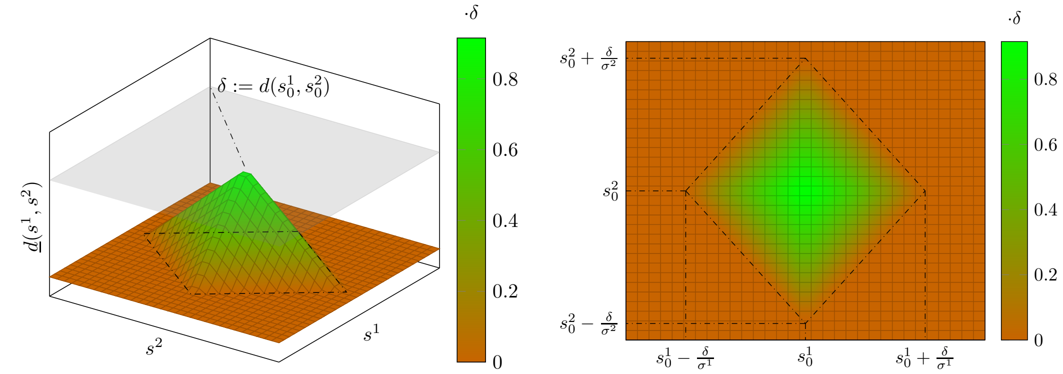
A bounding volume (BV) is used to contain sets of geometric primitives. A tree of BVs is used to model to the desired level of accuracy. Children of a BV contain partitions of the parent BV and the leaf nodes of the tree contain one primitive; a triangle, for example.

The collision query starts by comparing the BV roots for each object. If these overlap, then the query is applied recursively to their children. Otherwise, if no overlap is present, then the algorithm returns stating that there is no collision. If a leaf node is involved, then a collision test is conducted on it directly. Exact and approximate distance computations proceed in similar fashion.

A special case of collision detection, peculiar to the subject of this thesis, is when two moving agents are tested against each other on fixed geometric paths but with uncertainty in their velocities or subject to unpredictable stops. This is typical for the robot motions generated by a robot controller executing program instructions, see (O'Donnell et al. 1989; Åblad, Spensieri, et al. 2021). Indeed, the geometric path when executing a motion instruction may be considered fixed. However, the robot might not have the nominal speed and, more importantly, is not controllable during the command execution by a mechanism/device external to the robot controller itself. Therefore, a more conservative approach must be considered to ensure that the parallel execution of the motions is collision-free. All possible configurations that two robots may assume between the start and end of a motion instruction must be considered. This may be done in several ways and this thesis has adopted a class of strategies to accomplish it, reliant on:

- distance computations between two robots at given configurations, plus
- *sensitivity* bounds on the distance they can move relative to the path parameterization, which might be deemed a velocity.

In Figure 2.7, it is possible to see the distance lower bound between two robots on their predefined paths or, in other words, the distance that any point on a robot can




(a) Distance lower bound $\underline{d}$, based on one distance $\delta$ and path sensitivities.

(b) Collision-free area in coordination diagram.

Figure 2.7: Distance lower bound and collision-free area between two robots on pre-defined paths.

move without colliding. This may be generated from a single distance computation $\delta$ at given robot configurations corresponding to the path parameterization $(s_0^1, s_0^2)$ and from the path *sensitivity* bounds $\sigma_1, \sigma_2$. The collision-free area in the CD is then given by $(s^1, s^2)$ satisfying $\sigma_1|s^1 - s_0^1| + \sigma_2|s^2 - s_0^2| < \delta$, see Figure 2.7b. For an overview and efficient ways of doing this, see (Åblad, Spensieri, et al. 2021).

Although powerful techniques may decrease computing times, the most expensive part is still collision testing. To cope with that, a very powerful practical approach is to delay collision test until it is really needed, see (Bohlin and Kavraki 2000; Bohlin 2001).

## 2.5 Lazy approach

The lazy strategy is a way of finding optimal solutions within reasonable time in those situations when, given all necessary information, checking whether a solution is admissible in its domain is computationally more expensive than finding the solution itself and sees the problem in an “optimistic” way.

The basic theory behind the lazy strategy lies in a general relaxation theorem. Consider an optimization problem and its relaxation in the forms below:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{s.t.} \quad & x \in D, \end{aligned} \tag{2.3}$$

$$\begin{aligned} \text{minimize} \quad & f_R(x) \\ \text{s.t.} \quad & x \in D_R, \end{aligned} \tag{2.4}$$

where $f_R(x) \leq f(x), \forall x \in D$ and $D \subseteq D_R$, the following two properties hold, see (Patriksson et al. 2013):

1. if problem (2.4) is infeasible, then so is problem (2.3);
2. if (2.4) has an optimal solution $x_R^*$ such that $x_R^* \in D$ and $f_R(x_R^*) = f(x_R^*)$, then $x_R^*$ is an optimal solution for (2.3).

The overall algorithm works iteratively. At each iteration, the relaxed optimization problem is solved. If there is no solution, then the instance is labeled as unsolvable, otherwise distance queries are executed to validate that the obtained solution is collision-free. If no collision is detected, the algorithm returns an optimal solution; otherwise, the model is augmented with the newly detected collisions and a new iteration is carried out. The flowchart for the lazy coordination is illustrated in Figure 2.8. At each iteration, the problem may become more complex due to the introduction of variables and constraints that model the additional collision zones detected.

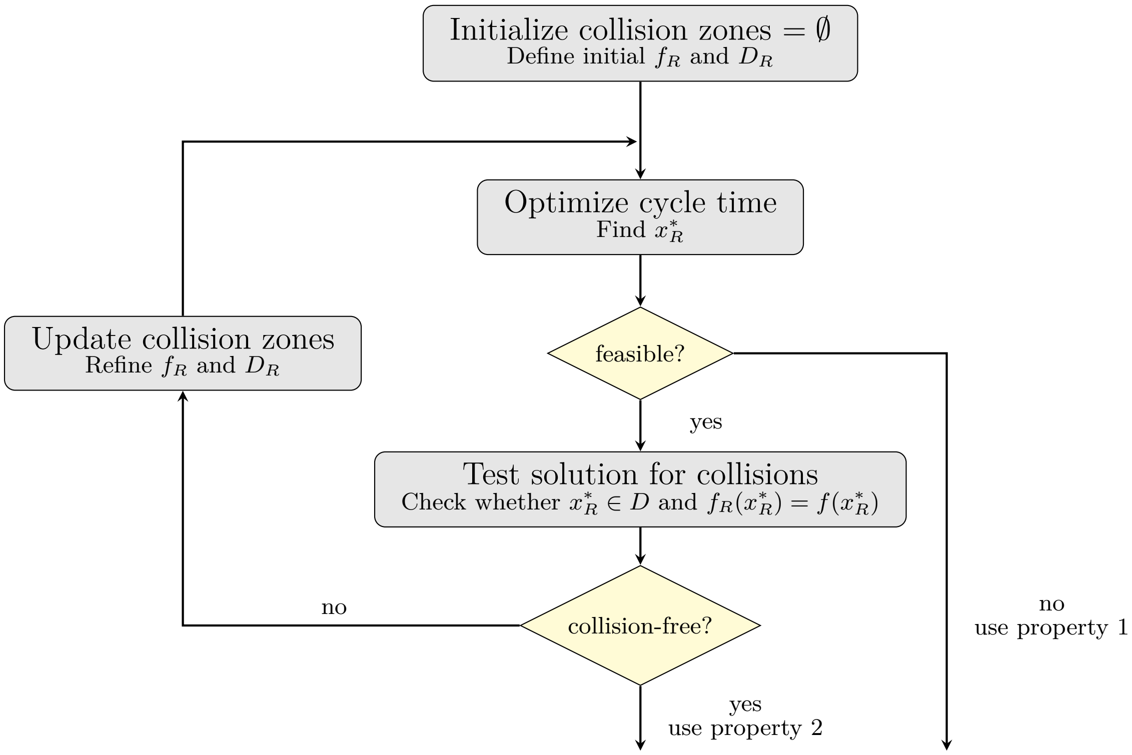



Figure 2.8: Flowchart for lazy optimization of cycle time in the path coordination problem.

The strategy used in the "Test solution for collisions" block is dependent on the problem characteristics. To minimize computing time, it aims to find a good trade-off between:

- being optimistic by searching small parts of the state space vs.
- gathering enough relevant information.

This approach appears in many of the algorithms included in this thesis. In one case, for example, the optimization step is the cycle time minimization for robot coordination. In another, it is the minimization of synchronization points. It is also used to optimize the sequence of tasks for a single robot (see Section 2.1.2), with unknown paths. The robot motion times are then relaxed based on their lower bound (defining $f_R(x)$) and with no collisions assumed (defining $D_R$). Robot path planning and distance queries are then carried out, refining $f_R$ and $D_R$.

# Chapter 3

# Contributions

This chapter describes the methods and algorithms designed and developed during the author's research activities and are organized according to the published scientific papers. However, these are not presented in chronological order. Rather they follow a logical thread which may aid reading and understanding of the rationale behind each method.

## 3.1 Summary of Paper A

*Collision-free coordination and visualization tools for robust cycle time optimization*

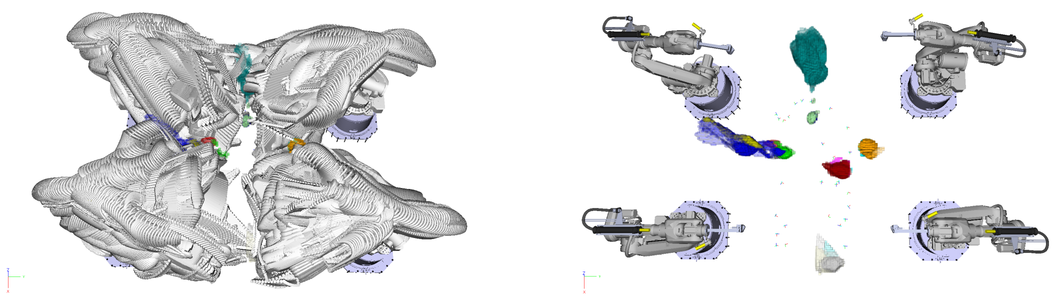
This article introduces the path coordination problem, highlighting current practice in industry and mainly addressing Research questions 2 and 3. The focus is not on completely new methods and algorithms but rather on presenting heuristics and strategies to support engineers in generating safe robot programs. Moreover, it compares two approaches to synthesize coordination schemas with respect to cycle time robustness.

The primary motivation was to investigate and provide some insights into the connection between intersection volumes in the 3D operational workspace (see Figure 3.1) and intersection zones in the generalized coordination diagram.

### 3.1.1 Contributions

The main contributions of this paper are:

- a comparison of how non-nominal robot motion times influence cycle time, in a permissive approach and in strict order of zones allocation;
- software tools to visualize intersection volumes and zones in the coordination diagrams;
- heuristics to minimize delays due to the implementation of synchronization mechanisms;
- the use of robots' sensitivity analysis to guarantee that two-robot motions are collision-free.



(a) Volumes swept by four industrial robots as they move.

(b) Multiple intersections for the swept volumes on the left.

Figure 3.1: Intersection volumes in the 3D work/operational space[1].

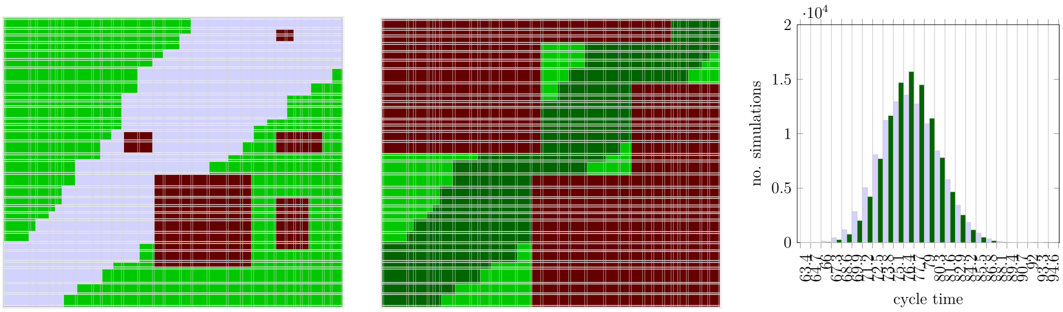
[1] from "An Iterative Approach for Collision Free Routing and Scheduling in Multirobot Stations", D. Spensieri, E. Åblad, J. Kressin, J. S. Carlson, A. Andersson, ASME Journal of Computing Science in Engineering, Vol. 21, No. 4, August 2021.

The paper describes a common way to produce interlocking policies in industry to avoid collisions between robots. It shows how guards are introduced to avoid these volumes and how deadlock can be detected. Moreover, it considers practical aspects such as: 1) the delays introduced when implementing such interlocking policies and 2) maintenance issues.

The current strategy adopted in industry is a “permissive” first-in-first-out (FIFO). Essentially, the idea is that a robot should not have to wait for another one if a shared volume is “free”, thus allocating it. The rationale for this behavior is that robots should not wait in case non-nominal robot motion times change the order that they enter the shared zones. An advantage of this strategy is that it is somewhat adaptive, relative to the non-nominal timing during the actual execution of the programs. The drawback is that a FIFO strategy for allocating a shared volume may lead to very high cycle times.

On the other hand, a “fixed” strategy (one that keeps the robot on the optimal schedule found by considering *nominal* robot motion times) will always let the robot allocate the shared zones in the same order. The drawback is that it is not reactive to non-nominal timing during the actual execution of the programs.

Naturally, a better strategy would be to include adaptive logic in the PLC, so that the best allocation order is implemented depending on the current state of the entire station. However, this strategy might place large memory requirements on the PLC and deadlock states must be identified. Furthermore, depending on the implementation, it might be necessary to update the PLC code any time robot programs are modified. This is not the case with the first two strategies (in which PLC code remains unchanged and coordination schemas are integrated in the robot programs) and therefore is not currently adopted in industry.




(a) Distribution of relative robot positions (cyan) in the FIFO approach.

(b) Distribution of relative robot positions (green) in the “fixed” approach.

(c) Cycle time distribution comparison for the FIFO and “fixed” approaches.

Figure 3.2: Distribution of relative robot positions and of cycle time in response to motion time variation[2].

[2]from “An Iterative Approach for Collision Free Routing and Scheduling in Multirobot Stations”, D. Spensieri, E. Åblad, J. Kressin, J. S. Carlson, A. Andersson, ASME Journal of Computing Science in Engineering, Vol. 21, No. 4, August 2021.

### 3.1.2 Results and discussion

The algorithms were included in a demonstrator version of the IPS simulation software. They were used to visualize the generalized coordination diagram and the intersection volumes for inspecting multi-robot stations in the automotive industry. The main idea was to support the engineers in achieving customized analysis and generating robot programs.

The same inspection instances were also used to carry out a study comparing a permissive FIFO approach to a “fixed” scheduling one. The experiments were run by introducing variation in robot motion times and show that the permissive approach might introduce a “combinatorial” effect, leading to very high cycle times. This drawback is avoided by using a fixed scheduling strategy, where the input variation shows a bounded effect on cycle time, equivalent to the single robot motion time variation. In some cases, however, they might work equally well, as the case of Figure 3.2.

But how can robot coordination, with its goal of optimizing cycle time, be done efficiently from a computational point of view? This question prompted the research work which led to the next article.

## 3.2 Summary of Paper B

*Coordination of robot paths for cycle time minimization*

Optimizing cycle time in a multi-robot station is a difficult task, due to the many variables and dofs involved. Some of the methods for finding feasible solutions rely on iterative algorithms, in which a main step is fixing the robot paths and tuning their velocities to avoid conflicts, i.e. the PCP, see Section 2.3. An efficient solver addressing this problem is proposed in the paper, which mainly investigated Research questions 1 and 2.

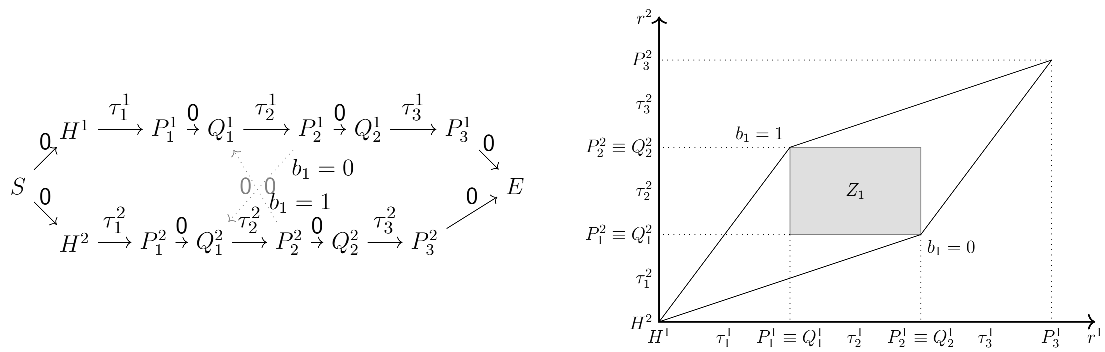
### 3.2.1 Contributions

The main contributions of this paper are:

- a model for the path coordination problem in a graph based way similar to job shop scheduling problem;
- a branch and bound algorithm solving the path coordination problem, exploiting the cylindrical structure of the problem.

A useful model used for job shop scheduling problem is the disjunctive graph, see (Jain et al. 1999). Here, each node models an operation belonging to a job; a set of conjunctive arcs, weighted with time, for example, connects consecutive adjacent nodes; a set of disjunctive arcs models the fact that one operation (head of the arc) may be done only after another one (tail of the arc). The minimum makespan can be found by searching for the longest path in the graph.

In the PCP, each node is a via-point in a robot path and the arc weights are motion times. A disjunctive arc is used to model the fact that a robot may enter a shared area only after another one has left the same area.




(a) Directed acyclic graph (DAG) view.

(b) Geometric 2D view.

Figure 3.3: Different views of a two-robot coordination problem[3].

[3]from "Coordination of robot paths for cycle time optimization", D. Spensieri, R. Bohlin, J. S. Carlson, IEEE International Conference on Automation Science and Engineering, Madison, USA, 2013.

In Figure 3.3 two different ways of modeling the problem are illustrated: Figure 3.3a shows a directed acyclic graph (DAG) view, whereas Figure 3.3b shows the corresponding visibility graph in the coordination diagram.

The problem may also be modeled using a mixed integer linear programing (MILP) formulation, see equations (3.1), with a boolean variable representing which robot enters the intersection zone first. This way of modeling collisions in multi-agent systems is mainly inspired by (Pallottino et al. 2002).

$$\begin{array}{rcl}
\text{minimize} & c & \text{s.t.} \quad \} \text{ objective} \\
\left.\begin{array}{r} c \geq \\ c \geq \end{array}\begin{array}{c} q_2^1 + \tau_3^1 \\ q_2^2 + \tau_3^2 \end{array}\right\} & & \text{makespan constraints} \\
\left.\begin{array}{r} h^1 \geq \\ p_1^1 \geq \\ q_1^1 \geq \\ p_2^1 \geq \\ q_2^1 \geq \\ h^2 \geq \\ p_1^2 \geq \\ q_1^2 \geq \\ p_2^2 \geq \\ q_2^2 \geq \end{array}\begin{array}{c} 0 \\ h^1 + \tau_1^1 \\ p_1^1 \\ q_1^1 + \tau_2^1 \\ p_2^1 \\ 0 \\ h^2 + \tau_1^2 \\ p_1^2 \\ q_1^2 + \tau_2^2 \\ p_2^2 \end{array}\right\} & & \text{priority constraints} \\
\left.\begin{array}{r} q_1^2 \geq \\ q_1^1 \geq \\ b_1 \in \end{array}\begin{array}{c} p_2^1 - Mb_1 \\ p_2^2 - M(1-b_1) \\ \{0,1\} \end{array}\right\} & & \text{mut.ex. constraints}
\end{array} \qquad (3.1)$$

In the suggested B&B algorithm, each node represents a sub-problem in which some of the disjunctive edges (relative to intersection zones) are present and the remainder are not.

Two child nodes are created by choosing an intersection zone and deciding that robot 1 enters the zone before robot 2 in one node and viceversa in the other node. Choosing a zone to branch on is done by estimating the one that mostly degrades cycle time. The bounding function used is the longest path on the DAG with the current arcs. This bound is very fast to compute and one may also exploit information from parent nodes to conduct incremental computations, see (Katriel et al. 2005). A second bound that is used is the cycle time for pairwise robots, computed in polynomial time by the A*. Another artifact used to speed-up the algorithm is trying to assign the remaining arcs at a certain node during the search, without increasing the node's lower bound. Thus, the entire branch may be pruned.

### 3.2.2 Results and discussion

The proposed B&B algorithm, with its different lower bounds, performs well compared to a general MILP solver package. Random instances have been generated to assess the correctness of the implementation and these show that computing times differ by several orders of magnitude in favor of the proposed algorithm. This is not unusual since this ad-hoc solver exploits the specific structure of the problem.

The algorithm has been included in a demonstrator version of the IPS software suite. Within this framework, an industrial test case from the automotive industry is solved and robot programs fulfilling the optimal schedule are generated by introducing synchronization signals. The efficiency of the solver and robustness of the generated robot programs make the method very appealing in practice.

However, in reality, these signals may cause that a robot stops because it needs to handshake with a PLC, thus introducing a delay not modeled during optimization. This implementation issue is considered in next article, which suggests a post-processing step to minimize the delays effect on cycle time.

## 3.3 Summary of Paper C

*Modeling and optimization of implementation aspects in industrial robot coordination*

This article considers the minimization of synchronization locations when robot programs are executed. In fact, such synchronization instructions usually introduce time delays, for two main reasons:

- robot stopping time at a location where synchronization is needed, due to dynamic effects;
- communication time with the PLC. With modern hardware, this second effect often has less influence than the first one.

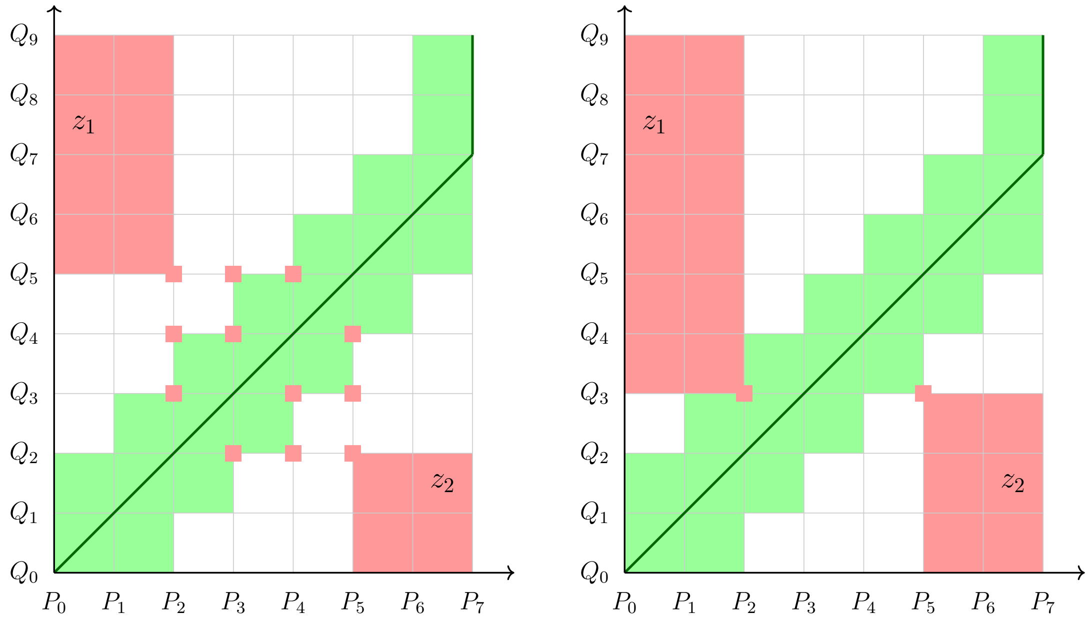
In certain industrial setups, even the number of ports/sockets allocated for such purposes in robot controllers might be limited. This provides another argument for the importance of the minimizing the number of variables used to model shared volumes. So, given the focus on practical issues, the paper mainly addresses Research question 3.

### 3.3.1 Contributions

The main contributions of the method proposed in this article are:

- a model for the time delays due to the introduction of synchronization instructions, with two different setups based on hardware implementation;
- a heuristic algorithm to minimize the number of synchronization locations in robot programs;
- solution of a case study in automotive inspection with the aim to validate and verify the proposed method.

Given a set of robots $\mathcal{R}$ and a set of via-points $\mathcal{V}^r$ for each robot $r$, a synchronization signal may only be fired at one of these locations, as they correspond to program instructions. Moreover, let $\mathcal{Z}$ be the set of intersection zones and the order in which the robots enter each zone fixed, then a zone may then be identified by one corner $(i^r, j^s)$ of the corresponding rectangle (northwest corner if the vertical robot enters the zone first, southeast otherwise) in the CD, see Figure 3.4. For each zone, the corner may be moved to many other locations, provided the optimal solution is not changed. One sufficient way to verify that is to let the chosen via-points avoid overlapping the cells occupied by the optimal solution. Now, let $\mathcal{I}_z$ identify the set of possible locations for a zone $z$, then the optimization problem to be solved is:




(a) Possible synchronization locations for two intersection zones.

(b) Three chosen synchronization locations: $P_2, P_5$ for robot 1, $Q_3$ for robot 2.

Figure 3.4: Coordination diagram with possible synchronization locations for two intersection zones.

$$\text{minimize} \quad \sum_{r\in\mathcal{R}} \sum_{i\in\mathcal{V}^r} x_i^r \tag{3.2a}$$

$$\text{s.t.} \quad \sum_{(i^r,j^s)\in I_z} x_i^r x_j^s \quad \geq 1, \qquad z \in \mathcal{Z} \tag{3.2b}$$

$$x_i^r \quad \in \{0,1\}, \qquad i \in \mathcal{V}^r, r \in \mathcal{R} \tag{3.2c}$$

Here, each $x_i^r$ takes the value 1 if the via-point $i$ for robot $r$ is used, otherwise 0. All intersection zones must be covered. This condition gives rise to constraints (3.2b), which basically state that at least one corner out of all possible pairs of via-points must be chosen for a zone. In Figure 3.4, for example, $\mathcal{I}_1 = \{(P_2, Q_3), (P_2, Q_4), (P_2, Q_5), (P_3, Q_4), (P_3, Q_5), (P_4, Q_5)\}$.

Note that this problem formulation is nonlinear due to the quadratic constraints (3.2b). In this paper, besides reformulating it with an MILP model, an efficient heuristic algorithm is proposed, acting directly on model (3.2). The algorithm works by maintaining a matrix $M$ which has rows representing the intersection zones and columns representing all possible combinations of via-points, *i.e.* $\bigcup_{z\in Z} I_z$. An entry in $M$ is 1 if the corresponding $(i^r, j^s)$ via-points combination covers the zone corresponding to the row, otherwise 0. It starts with an iterative construction of an initial set of via-points, covering all zones. During this phase, the matrix $M$ is updated by removing the chosen via-points and merging columns corrsponding to identical via-points. An improvement phase is then run. This is done by removing some of the chosen via-points and trying to replace them, the aim being to decrease the objective function.

### 3.3.2 Results and discussion

From a computational point of view, the proposed heuristic algorithm performs very well in terms of quality of the solution obtained and computing time (when compared to a general purpose MILP solver). The method has been integrated with the IPS software suite and applying it to industrial test cases allows the reduction of synchronization points, with consequent cycle time improvement.

The objective function used in this article is an attempt to decrease the time delays introduced by synchronization instructions in robot programs, necessary to avoid collisions. At any rate, minimizing the total number of locations in which communications occur does not really guarantee that the best cycle time will be achieved. Indeed, it would be more appropriate to have a min-max model and even better modeling the delays in the original coordination problem and not just as a post-processing step. However, despite being highly relevant improvements, these have not yet been investigated (either in this paper or in this thesis) due to the greater complexity of modeling and solving the resulting problems.

Furthermore, do we actually need synchronization signals in typical robot stations, to satisfy the cycle time requirements imposed by the production/inspection process? This question is answered in next article.

## 3.4 Summary of Paper D

*Intersection-free geometrical partitioning of multirobot stations for cycle time optimization*

This article presents a method aiming to generate robot paths, when possible, completely disjoint from each other. From a practical point of view, this means that the corresponding robot motions would never collide with each other, independently of their timing, thus without needing synchronization instructions. Therefore, it mainly addresses Research questions 1 and 3.

### 3.4.1 Contributions

The method tackles the overall problem by dividing it in several sub-problems that are solved successively and iterated in a lazy fashion, once more information has been gathered from simulations.

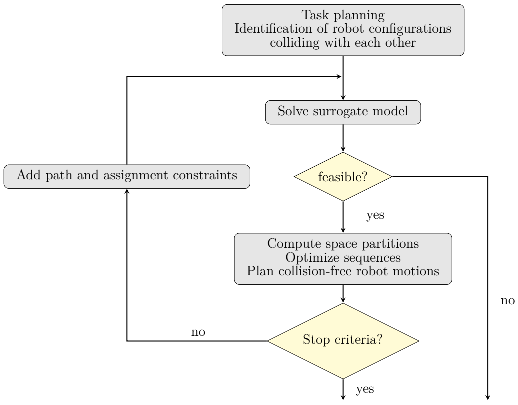
The cycle time minimization problem is approximated by a simplified model neglecting motion times but only considering process times (such as welding or inspection using a camera or laser sensor). This is more tractable from the computational point of view. The assumption is also motivated by the fact that, in many applications, the total process time is comparable to or even dominates the motion times between process locations.

The main contributions of the paper are:

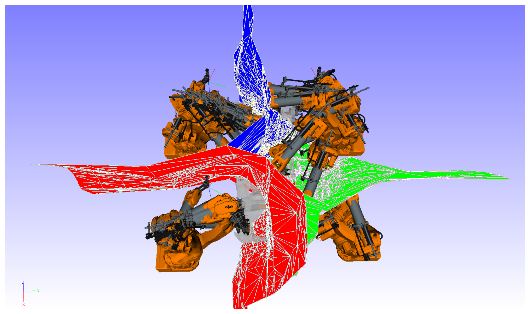
- an iterative method to generate robot motions geometrically disjoint from each other;
- a surrogate model for minimization of cycle time with conflicting robot configurations;
- generation of triangulated surfaces to delimit the workspaces corresponding to robot partitions, by approximating the Generalized Voronoi Diagram;
- solution of case studies in the automotive industry with the aim to validate and verify the proposed method.

The method starts with a pre-processing step, not included in the loop, which carries out *task planning* for each task $i \in \mathcal{T}$, see Section 1.3: for each robot $r \in \mathcal{R}$ and task $i$ it computes a set $\mathcal{F}^r(i)$ of configurations that match the task definition. This step also generates a set $\mathcal{D}$ of pairs of robot configurations $(i_1 j_1 r_1, i_2 j_2 r_2), i_1 \neq i_2, r_1 \neq r_2$ that do not satisfy mutual distance requirements.

The main step, partitioning the tasks, is to first design a mathematical model that minimizes the makespan and accounts for the restrictions found:




(a) Flowchart for the generation of intersection-free robot motions .



(b) Robot configurations at assigned tasks with surfaces separating their workspaces[4].

Figure 3.5: Flowchart for the method devised in this paper and example illustrating the main idea.

$$
\begin{aligned}
&\text{minimize} \quad c && && \text{(3.3a)}\\
&\text{s.t.} \sum_{i\in\mathcal{T}} \sum_{j\in\mathcal{F}^r(i)} c_{ij}^r x_{ij}^r \quad \leq \quad c, && r \in \mathcal{R} && \text{(3.3b)}\\
&\sum_{r\in\mathcal{R}} \sum_{j\in\mathcal{F}^r(i)} x_{ij}^r \quad = \quad 1, && i \in \mathcal{T} && \text{(3.3c)}\\
&x_{i_1j_1}^{r_1} + x_{i_2j_2}^{r_2} \quad \leq \quad 1, && (i_1j_1r_1, i_2j_2r_2) \in \mathcal{D} && \text{(3.3d)}\\
&x_{ij}^r \quad \in \quad \{0,1\}, && j \in \mathcal{F}^r(i), i \in \mathcal{T}, r \in \mathcal{R}. && \text{(3.3e)}
\end{aligned}
$$

A solution to this model may be found by using MILP solver packages and may be written in another form to obtain faster computations, as in the paper. It essentially generates a set of configurations for each robot, covering all tasks, without any pairwise collisions.

The geometric information about these robot configurations is then used to generate surfaces identifying disjoint space volumes, possible if the model is feasible. An example of robot configurations generating separating surfaces is shown in Figure 3.5b.

The cycle time estimation is then refined by considering point-to-point motion planning and sequencing of tasks (GTSP) for each robot, without the need to change the load or check for collisions between robots. This step is done by using existing built-in functionalities in the IPS software suite. If the solution found meets the stop criteria, then the algorithm terminates. Otherwise, a new iteration is carried out, by excluding similar solutions or by including information on problematic motions, depending on whether cycle time requirements have not been met or collision-free motions are difficult to obtain. The steps are illustrated in Figure 3.5a.

[4]from "Intersection-Free Geometrical Partitioning of Multirobot Stations for Cycle Time Optimization", E. Åblad, D. Spensieri, R. Bohlin, J. S. Carlson, IEEE Transactions on Automation Science and Engineering, Vol. 15, No. 2, pp. 842-851, April 2018.

Note that the algorithm may be even applied to robot lines, consisting of multi-robot stations placed in a line and disjoint from each other, still sharing one set of tasks to be distributed among all robots. In this case, one can refine cycle time estimation by solving a min-max heterogeneous GVRP, without considering collisions between robots.

### 3.4.2 Results and discussion

Thanks to the integration with the IPS software platform, the method has been applied to several classes of multi-robot stations, specifically single and multi-station (line) stud welding applications, and single station inspection processes. The size of the instances varied from four to ten robots conducting a maximum of about 200 tasks. Well-balanced tasks partitions could be found in all of them, with cycle times comparable to the solutions obtained by other optimization methods that allow time synchronization among robots. Scenarios in which this approach is unsuitable have also been identified.

Keeping the cycle time under a specific threshold for some typical stations is doable, especially if there is a considerable freedom as to how tasks may be distributed and provided the robots are quite well separated from each other (as is the case for lines of serial stations).

Sometimes, not only is it impossible to satisfy the cycle time, there may not be any solution that has completely disjointed robot paths. A natural question, therefore, is whether parts of the paths may be allowed to overlap in space, while altering the timing and order of tasks so as to avoid collisions. This approach is investigated in the next article, which provides solutions to problems in which robots have small clearance to the environment and between each other.

## 3.5 Summary of Paper E

*An iterative approach for collision free routing and scheduling in multi-robot stations*

This article mainly addresses Research question 1. It describes a method for avoiding collisions between robots and minimizing cycle time for multi-robot stations, by changing the way in which their operations/tasks are scheduled and by velocity-tuning their motions.

An initial distribution of tasks between the robots must be provided: this assignment might be set by the manufacturing engineer or may be the result of a previous optimization step. In the article, this assignment is generated by a direct B&B method and constitutes the starting point for running synchronous scheduling.

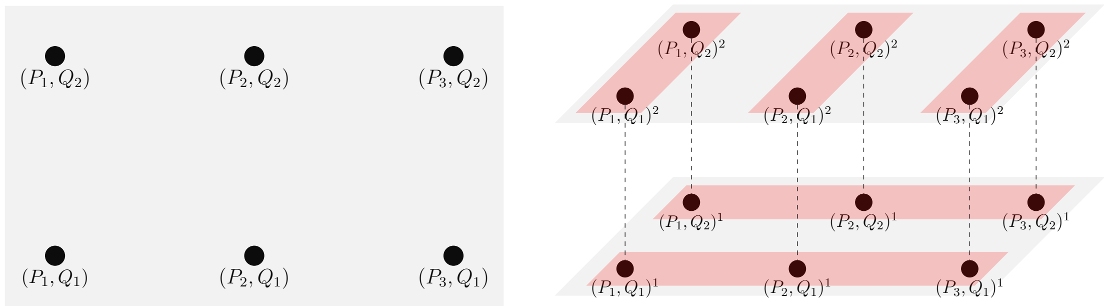
### 3.5.1 Contributions

The highlights of this paper are:

- a branch and bound algorithm for a min-max heterogeneous GVRP, without conflicts;
- a model assuming synchronous robot motions to resolve collisions between robots;
- a novel transformation of the proposed model into a generalized traveling salesman problem.

The min-max heterogeneous GVRP is approached using a direct B&B technique. The branching step consists of choosing a task and creating $N_R$ child nodes (with $N_R$ being the number of robots), each assigning the task to a specific robot. Thus, at each node of the tree, we have $N_R$ sets $G^r$ (possibly empty) for each robot $r$, with the respective tasks so far assigned. The bounding procedure consists of solving a GTSP for each robot, defined by the set $G^r$. This bound may be tightened by using the triangle inequality and exploiting the fact that the time to carry out a task is usually of the same order of magnitude as the motion times between tasks.

Once the robot loads are fixed, a second algorithm is proposed to avoid collisions between robots. Essentially, a synchronized state space, $S_G = G^1 \times \ldots \times G^{N_R}$ is constructed to transform the timing aspect in a combinatorial fashion. Naturally, the simplification made involves the movements between robot tasks being synchronous. Moreover, to capture the fact that all tasks must be done at least once, the state space is cloned for each robot and partitioned into groups, as per GTSP. Figure 3.6 shows the artificial GTSP created for two robots, assigned two and three tasks respectively.




(a) Synchronized state space with 3 and 2 locations for robots 1 and 2, respectively.

(b) State space enhanced with artificial cloned states and GTSP groups (in red).

Figure 3.6: The synchronized state space with its clone and GTSP formulation groups.

### 3.5.2 Results and discussion

The load distribution algorithm (*i.e.* without considering collisions between robots) has been applied to geometric instances (from the TSPLIB) by introducing four fictitious robots. Three different strategies have been tested and the computational experiments show that it is possible to solve to optimality instances up to 40 tasks. Anyway, more powerful methods are already available, thus the algorithm has been provided because an initial assignment was needed and since it is easy to implement and does not rely on linear programming solvers.

The main purpose of the previous step was to provide a good initial distribution. The synchronous routing and scheduling algorithm, described in Section 3.5.1, was then also applied to some of the previously mentioned instances. The overall iterative method proposed was interfaced with the IPS software suite and applied to cases from the automotive industry. In these cases it was possible to improve cycle time by eliminating long wait times due to poor task sequencing choices. Therefore, this strategy might be used to try and obtain different sequencing and scheduling alternatives, especially for cases with small clearance, in which robot collisions require very long wait times. In other cases, when robot paths do not have large overlapping volumes, the initial solutions may already be of sufficient quality.

## 3.6 Summary of other relevant publications

The following publications are not included as formal contributions in the thesis but are relevant to mention. If one ordered all the thesis' papers, based on increasing level of generality, these additional papers would be positioned as first and last one: the first, positioned at a detail level, about how to efficiently verify that two robots satisfy distance requirements whilst moving along their assigned paths, whereas the second being an attempt to solve the general min-max heterogenous GVRP with conflicts.

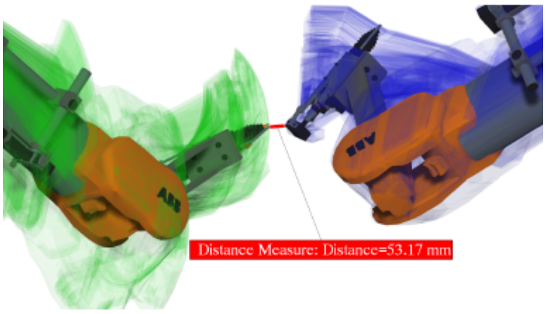
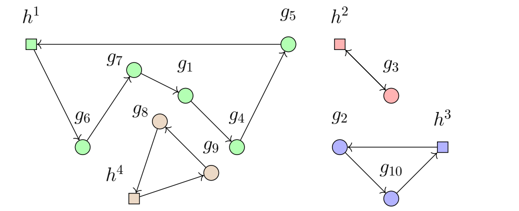



Figure 3.7: Minimum distance between two robots moving along their paths[5].




| $g_1$ | $g_2$ | $g_3$ | $g_4$ | $g_5$ | $g_6$ | $g_7$ | $g_8$ | $g_9$ | $g_{10}$ |
|---|---|---|---|---|---|---|---|---|---|
| 0.16 | 0.65 | 0.3 | 0.2 | 0.23 | 0.1 | 0.15 | 0.9 | 0.85 | 0.7 |
| 0.1 | 0.8 | 0.2 | 0.3 | 0.3 | 0.1 | 0.5 | 0.4 | 0.5 | 0.5 |

Figure 3.8: Chromosome encoding with corresponding robot routes.

### *Continuous collision detection of pairs of robot motions under velocity uncertainty.*

This paper, (Åblad, Spensieri, et al. 2021), proposes an efficient algorithm to check whether or not two robot paths always satisfy a minimum distance requirement, independently of their velocity (see Figure 3.7). It minimizes a function of the lower bound of two-robot distance, by assuming that a Lipschitz constant exists. At each iteration, it computes the actual distance between the robot configurations that minimize the lower bound, according to the current gathered information.

The algorithm proved highly efficient and suitable for use in solving path coordination problems and similar ones.

### *Throughput maximization by balancing, sequencing and coordinating motions of operations in multi-robot stations.*

This paper, (Spensieri et al. 2010), is an attempt to tackle the overall problem (see Section 2.2.2) of cycle optimization for multi-robot stations by balancing the tasks, sequencing them, and coordinating the robots' motions. The problem is solved through a genetic algorithm. A random-key encoding is used for a chromosome (see Figure 3.8), essentially stating which robot and configuration is chosen to perform

[5]from (Åblad, Spensieri, et al. 2021).

each task. The fitness evaluation is done by planning the motions between consecutive configurations and coordinating them. In addition to basic crossover and mutation, local optimization (based on GTSP algorithms) is carried out on each individual in a generation, at a given frequency.

This method was capable of addressing industrial instances with up to four robots and eighty tasks.

# Chapter 4

# Discussion of research approach

## 4.1 DRM evaluation

### 4.1.1 DRM steps

The building blocks of DRM were iterated during the research and it is sometimes hard to draw sharp boundaries between them. They are briefly summarized below.

**Research clarification:** the relevant problems were identified and defined in more detail. The success criteria were also initially defined as being able to automatically obtain high-quality (for example, with respect to cycle time and geometric robustness) solutions for industrial problems and decreasing commissioning time for the engineers, see Section 1.2.1.

**Descriptive study I:** literature studies were carried out and research and industrial gaps identified.

**Prescriptive study:** existing theories were adopted and further developed, plus new models have been devised for some of the problems encountered. Algorithms were implemented both at a prototype and more release-ready level.

**Descriptive study II:** the implemented methods were evaluated against literature, industrial and randomly generated cases. For some of the algorithms a comparison with existing software libraries (able to solve the same problems) was also carried out.

### 4.1.2 Evaluation of success criteria

The following is a brief discussion of the success criteria identified and formulated in Section 1.3.2.

**High-quality solutions:** the methods were integrated into demonstrator versions of the IPS software, to handle complex scenarios. The problem instances investigated have been in the order of two to four robots and forty tasks, with CAD models consisting also of millions of triangles. For many problems, solutions were not available previously, due to the novelty of the problem formulations. Therefore, achieving a solution fulfilling the requirements was positively evaluated. In those cases in which other solvers were available, the solutions were compared. The results showed that exact solutions were achieved for small- to medium-size problems. For larger problems, for which optima were not available, the proposed heuristic methods provided solutions between the lower and upper bounds computed by the same software packages.

**Solving speed:** in all appended papers, the proposed methods generally provide feasible solutions when integrated with the IPS software, in terms of minutes/hours. For many problems, solutions were not available previously, due to the novelty of the problem formulations. Therefore, achieving a solution fulfilling the requirements was positively evaluated. In those cases in which other solvers were available, the solutions were compared, mostly to MILP package solvers (see (Forrest et al. 2005)). The results showed that exact solutions were achieved in equivalent or lower running times, whereas, for large instances, hardly solvable exactly, heuristic approaches provided solutions in times comparable to the same packages.

**Decrease commissioning time:** the methods were integrated into demonstrator versions of the IPS software, with appropriate graphical user interface, to facilitate the use by expert engineers in the field. Automatic robot program generation and automatic coordination scheme export were also provided. This was a contribution to prove how virtual product and production realization may be achieved. Indeed, (IPS 2017) shows proven 75% faster commissioning.

**Scientific contribution:** all five appended papers contribute to answer the three research questions formulated in Section 1.3.1. The articles were shared with the scientific community, through publication in peer-reviewed international journals and at a conference.

### 4.1.3 Validation and verification

To provide more evidence for the *validation* process, it has been advantageous in this thesis to implement the algorithms and methods in high-level methods that are closer to the end-user. Indeed, this made it easier to evaluate whether the method was doing what it should. In cases when a previous functionality was upgraded by a new algorithm, the newly developed method inherited the validated properties of the previous one. If a new functionality was created, it was then included in demonstrator versions and applied to industrial test cases.

The *verification* process (ensuring that the methods developed satisfied the relevant requirements and specifications) was done by applying the methods to various industrial and randomly generated instances and by checking that the solutions obtained were feasible. Where possible, the results were compared with other algorithms solving the same problem. This helped support the evidence that "the system was built right". A further point for *verify* the validity of the methods was *acceptance* of their description in international peer-reviewed journals and conferences. This shows that experts in relevant fields deemed the results of this research to be a contribution to the scientific community.

## 4.2 WQL research strategy evaluation

### 4.2.1 Answering research questions

The research questions, see Section 1.3.1, have been discussed throughout this work by highlighting how each appended paper has contributed to answering them. The papers' contributions are shown below for each of the questions.

**Research question 1**

How might the equipment utilization of multi-robot stations be improved?

All the papers have contributed by optimizing specific aspects of the current processes. Paper A provided heuristics for optimizing cycle time by introducing synchronization signals, describing both an *operational space intersection zone* approach and a *coordination space intersection zone* one. It also provided support, in the form of software tools, to engineers during planning the manufacturing/inspection processes. Paper B enabled cycle time minimization by providing a highly efficient algorithm for use as a building block of methods solving more complex problems such as that in Paper E. Indeed, this contribution provided methods for obtaining low cycle times by optimizing the order of tasks and configurations chosen by each robot, initially neglecting collisions. It then provided an algorithm to resolve collisions between robots using a synchronous scheduling algorithm; something that was particularly useful in very cluttered environments.

**Research question 2**

How might robots avoid mutual collisions when executing geometrically predefined robot programs, so as to minimize cycle time?

Paper B provided a new algorithm for minimizing cycle time, based on branch and bound on a MILP formulation and exploiting different views of the problem: the disjunctive graph model and the geometric model for the two-robots case. Moreover, it was included in a method for handling complex scenarios in which collision testing constitutes a bottleneck. Paper A also dealt with this question and aimed to coordinate robots when the scheduling was not fixed. The paper also highlighted the differences from a fixed strategy. These methods are particularly useful when robot motions are difficult to obtain, as in complex spot-welding applications, or when there is a need to follow a pre-defined curve with a tool center point (TCP).

Indeed, these restrictions offer fewer opportunities for using the robot's dofs to avoid collisions.

**Research question 3**

How might relevant industrial implementation requirements be modeled to optimize cycle time?

The "more practical" aspects considered here were the robot program maintenance issues and the time delays introduced by the current way of implementing robot interlocking policies. Paper A has considered a way to work in practice where sub-programs are glued together into a full one, limiting maintenance of such programs by the engineers. In Paper D the need for synchronization was completely avoided by using a method that generates robot motions which are entirely disjointed from each other. Paper C modeled these delays explicitly and provided an algorithm for their minimization as a post-processing step, once a global solution (neglecting the delays) had been found. Paper A also contributed by suggesting heuristics to achieve "larger" intersection zones, with the aim to synchronize robots in a small number of locations.

### 4.2.2 Scientific and industrial contributions

#### Scientific contributions

The scientific contributions, highlighted throughout this thesis, are here briefly summarized.

**Itersection zones:** insights about the relation between intersection zones in the operational workspace and coordination space. An initial study has been conducted about how non-nominal robot motion times influence cycle time, depending on the adoption of a permissive FIFO zone allocation strategy vs. a fixed-order one.

**Path coordination:** a novel and efficient algorithm to coordinate robot paths by velocity tuning and by introducing synchronization instructions in practice.

**Syncronization minimization:** a novel method to minimize the contribution of the synchronization instructions to cycle time.

**Intersection-free:** a novel method to cover pre-defined robot tasks in minimal time, by generating robot motions, completely disjoint from each other.

**Routing and scheduling:** a novel method to synchronize robot motions by changing task ordering and timing.

### Industrial contributions

The industrial contributions, highlighted throughout this thesis, are here briefly summarized.

**Current practice** in industry has been taken considerably into account, by:

- modeling specific industrial aspects;
- proposing an incremental optimization of the currently adopted methods in industry;
- comparing it to the newly proposed methods.

**General methods** have been proposed with the possibility of being applied to a variety of processes such as welding, sealing, inspection and others.

**Demonstrator** versions of the software IPS have integrated the proposed methods and algorithms, thus allowing their application to real world scenarios. These include multi-robot stations consisting of millions of triangles, with two to four robots and up to forty tasks (more for some of the methods). By doing that:

- production engineers can benefit of an arsenal of methods that can be used to address the challenges arising in several processes and different station layouts. For example, it might be possible to generate more efficient coordination schemas on already existing robot programs, or optimize cycle time for multi-robot stations even by generating new robot motions. It might even be possible to use the proposed methods at an early stage, when designing the layout of new production cells.
- it might be possible to design robot motions that fulfill cycle time requirements, thus eliminating one robot, for example. Or there might be the possibility to increase the throughput, while still keeping the same equipment.
- they contributed, to an extent that unfortunately is difficult to quantify, to achieve 75% faster commissioning, as stated in (IPS 2017).

**Preliminary industrial study:** currently used robot programs in inspection applications have been simulated in IPS and cycle time reduced by 15%, after re-ordering the tasks and introducing new coordination schemas. The actual time estimation from the real station is currently under investigation.

# Chapter 5

# Conclusions and future work

## 5.1 Conclusions

This thesis resumes the work done of recent years on planning multi-robot stations in industrial environments, focusing on robot coordination for collision avoidance and cycle time optimization. The scope of this thesis was to provide algorithms, methods and tools to support engineers in the process of virtual product and production realization, as in the spirit of Wingquist Laboratory. Indeed, industrial needs and research challenges were the main drivers for the work, with either the first or second aspects sometimes taking the major role.

The thesis contributed to knowledge by answering three research questions, regarding:

- improvement of equipment utilization in multi-robot stations;
- mutual collision avoidance and cycle time minimization for predefined robot paths;
- formulating and integrating relevant industrial aspects into optimization models.

To accomplish that, five research papers were included, each focusing on a single theme or on a combination of them.

The main research challenge was to produce an efficiently integrated model of the combinatorial aspect of the problems alongside the geometrical and time aspects. Moreover, attention was given to the industrial requirements, so as to capture these aspects within a model suitable for optimization.

A main enabler for the research has been the existence of offline simulation software, providing a platform with many of functionalities needed to investigate complex scenarios, in particular robot applications in automotive, such as assembly, sealing and inspection.

Based on that, implementing the contributions and results of this research into software demonstrators is believed to shorten the time between planning in virtual environments and implementation in reality. Nevertheless, such methods are believed to serve as a powerful tool for evaluating different production designs and improving existing solutions.

Moreover, it is important to place many of the algorithms and methods in the right time frame and to understand their motivations. Indeed, it has been challenging to refer to similar works in the research literature, due to this interdisciplinary field being quite an emerging area. This is especially true considering the complexity of the target scenarios, in terms of CAD models size, number of robots and tasks, and no less in view of the current industrial setup being able to consider new solutions, outside specific standards.

A lot has happened during the last ten years, regarding computer hardware development and availability of software libraries, for sure moving this research field to a faster-growing phase.

## 5.2 Future work

The study of such an interdisciplinary problem offers many opportunities for improvement and raises questions for further investigation.

**Improving sub-problems solution:** in this perspective, it is worth studying how to decompose the problem in different ways and get better insights into the natural difficulties of each aspect of the overall problem. For example:

- coordinating robot motions by using roadmaps or more degrees of freedom;
- checking for fulfillment of several different clearance requirements;
- improving performance issues in combinatorial problems arising in the model and enabling larger instances to be solved;
- introducing additional constraints in the sub-problems, such as priority constraints, to capture further industrial processes.

**Solving the overall problem:** integrating all the sub-problems into a global optimization method remains a challenge, especially when it concerns gaining a better understanding of how to integrate time into geometric aspects. In this regard, a further avenue of investigation might be to determine the models most suitable for combining and then to identify the dofs with the greatest impact on cycle time optimization. Moreover, with the development of computer hardware and with the availability of software tools, path planning in high-dimensional spaces might be approached in a more direct way.

**Human-robot collaboration:** on the other hand, in facing the new challenges brought by the growth of collaborative robots applications, directions for further research might be:

- balancing operations between humans and robots, based on ergonomics and cycle time;
- coordinating robot motions operating in the same environment as humans.

# Bibliography


Åblad, E., D. Spensieri, R. Bohlin, and A.-B. Strömberg (2021). “Continuous Collision Detection of Pairs of Robot Motions Under Velocity Uncertainty”. In: *IEEE Transactions on Robotics*, pp. 1–12. DOI: 10.1109/TRO.2021.3050011 (cit. on pp. 22, 23, 40).

Åblad, E., A.-B. Strömberg, and D. Spensieri (2021). “Exact makespan minimization of unrelated parallel machines”. en. In: *Open Journal of Mathematical Optimization* 2, 2. DOI: 10.5802/ojmo.4 (cit. on p. 17).

Applegate, D. L., R. E. Bixby, V. Chvatal, and W. J. Cook (2019). “A Dynamic Programming Approach to Sequencing Problems”. In: (cit. on p. 14).

Baldacci, R., M. Battarra, and D. Vigo (Jan. 2008). “Routing a Heterogeneous Fleet of Vehicles”. In: vol. 43, pp. 3–27. ISBN: 978-0-387-77777-1. DOI: 10.1007/978-0-387-77778-8_1 (cit. on p. 16).

Blessing, L. T. M. and A. Chakrabarti (2009). *DRM, a Design Research Methodology*. 1st. Springer Publishing Company, Incorporated. ISBN: 1848825862 (cit. on pp. 5, 6).

Boehm, W. W. (1979). “Editorial”. In: *International Social Work* 22.1, pp. 1–1. DOI: 10.1177/002087287902200101. eprint: https://doi.org/10.1177/002087287902200101 (cit. on p. 6).

Bohlin, R. (2001). “Path planning in practice; lazy evaluation on a multi-resolution grid”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2001: Expanding the Societal Role of Robotics in the the Next Millennium, Maui, HI, USA, October 29 - November 3, 2001*, pp. 49–54. DOI: 10.1109/IROS.2001.973335 (cit. on p. 23).

Bohlin, R. and L. E. Kavraki (2000). “Path Planning Using Lazy PRM.” In: *ICRA*. IEEE, pp. 521–528. ISBN: 0-7803-5889-9 (cit. on p. 23).

Canny, J. F. (1988). *The Complexity of Robot Motion Planning*. Cambridge, MA, USA: MIT Press. ISBN: 0-262-03136-1 (cit. on p. 21).

Cantos Lopes, T., C. G. S. Sikora, R. Gobbi Molina, D. Schibelbain, L. C. A. Rodrigues, and L. Magatão (2017). “Balancing a robotic spot welding manufacturing line: An industrial case study”. In: *European Journal of Operational Research* 263.3, pp. 1033–1048. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2017.06.001 (cit. on p. 8).

Carlson, J. S., D. Spensieri, K. Wärmefjord, J. Segeborn, and R. Söderberg (2014). “Minimizing Dimensional Variation and Robot Traveling Time in Welding Stations”. In: *Procedia CIRP* 23. 5th CATS 2014 - CIRP Conference on Assembly

Technologies and Systems, pp. 77–82. ISSN: 2212-8271. DOI: `https://doi.org/10.1016/j.procir.2014.03.199` (cit. on pp. 13, 15).

Ekstedt, F. and D. Spensieri (2009). *A direct Lin-Kernighan heurtistic for the generalized traveling salesman problem*. Tech. rep. Fraunofer-Chalmers Centre (cit. on p. 15).

Escudero, L. F. (1988). "An inexact algorithm for the sequential ordering problem". In: *European Journal of Operational Research* 37.2, pp. 236–249. ISSN: 0377-2217. DOI: `http://dx.doi.org/10.1016/0377-2217(88)90333-5` (cit. on p. 14).

Fischetti, M., J. J. Salazar González, and P. Toth (June 1997). "A Branch-and-Cut Algorithm for the Symmetric Generalized Traveling Salesman Problem". In: *Operations Research* 45, pp. 378–394. DOI: `10.1287/opre.45.3.378` (cit. on p. 15).

Forrest, J. and R. Lougee-Heimer (2005). "COIN Branch and Cut (CBC) solver". In: (cit. on p. 44).

Garey, M. R. and D. S. Johnson (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co. ISBN: 0716710455 (cit. on p. 13).

Ghiani, G. and G. Improta (2000). "An efficient transformation of the generalized vehicle routing problem". In: *European Journal of Operational Research* 122.1, pp. 11–17. ISSN: 0377-2217. DOI: `https://doi.org/10.1016/S0377-2217(99)00073-9` (cit. on p. 16).

Gutin, G. and A. P. Punnen (2002). *The Traveling Salesman Problem and Its Variations*. Combinatorial Optimization. Springer US. ISBN: 9781402006647 (cit. on p. 14).

Held, M. and R. M. Karp (1961). "A Dynamic Programming Approach to Sequencing Problems". In: *Proceedings of the 1961 16th ACM National Meeting*. ACM '61. New York, NY, USA: Association for Computing Machinery, pp. 71.201–71.204. ISBN: 9781450373883. DOI: `10.1145/800029.808532` (cit. on p. 13).

Held, M. and R. M. Karp (Dec. 1971). "The Traveling-Salesman Problem and Minimum Spanning Trees: Part II". In: *Math. Program.* 1.1, pp. 6–25. ISSN: 0025-5610. DOI: `10.1007/BF01584070` (cit. on p. 14).

Helsgaun, K. (2006). *An effective implementation of k-opt moves for the LinKernighan TSP heuristic*. Tech. rep. Roskilde University (cit. on p. 14).

Helsgaun, K. (Sept. 2015). "Solving the equality generalized traveling salesman problem using the Lin–Kernighan–Helsgaun Algorithm". English. In: *Mathematical Programming Computation* 7.3, pp. 269–287. ISSN: 1867-2949. DOI: `10.1007/s12532-015-0080-8` (cit. on p. 15).

Hömberg, D., C. Landry, M. Skutella, and W. Welz (2017). "Automatic reconfiguration of robotic welding cells". In: *Math for the Digital Factory*. Springer, pp. 183–203 (cit. on p. 3).

"IEEE Standard for Software Verification and Validation" (1998). In: *IEEE Std 1012-1998*, pp. 1–80. DOI: `10.1109/IEEESTD.1998.87820` (cit. on p. 7).

IPS (2017). "IPS Robot Optimization". In: (cit. on pp. 44, 47).

Isto, P. and M. Saha (June 2006). “A slicing connection strategy for constructing PRMs in high-dimensional cspaces”. In: vol. 2006, pp. 1249–1254. DOI: 10.1109/ROBOT.2006.1641880 (cit. on p. 17).

Jain, A. S. and S. Meeran (1999). “Deterministic job-shop scheduling: Past, present and future”. In: *European Journal of Operational Research* 113.2, pp. 390–434. ISSN: 0377-2217. DOI: https://doi.org/10.1016/S0377-2217(98)00113-1 (cit. on p. 29).

Karaman, S. and E. Frazzoli (2011). “Sampling-based algorithms for optimal motion planning”. In: *I. J. Robotics Res.* 30.7, pp. 846–894 (cit. on p. 21).

Karapetyan, D. and G. Gutin (2011). “Lin–Kernighan heuristic adaptations for the generalized traveling salesman problem”. In: *European Journal of Operational Research* 208.3, pp. 221–232. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2010.08.011 (cit. on p. 15).

Katriel, I., L. Michel, and P. Hentenryck (Apr. 2005). “Maintaining Longest Paths Incrementally”. In: *Constraints* 10.2, pp. 159–183. ISSN: 1383-7133. DOI: 10.1007/s10601-005-0554-9 (cit. on p. 30).

Kavraki, L. E., P. Svestka, J.-C. Latombe, and M. Overmars (1996). “Probabilistic Roadmaps for Path Planning in High Dimensional Configuration Spaces”. In: *IEEE Transactions on Robotics and Automation* 12.4, pp. 566–580. DOI: 10.1109/70.508439 (cit. on p. 21).

Landry, C., R. Henrion, D. Hömberg, M. Skutella, and W. Welz (2013). “Task assignment, sequencing and path-planning in robotic welding cells”. In: *2013 18th International Conference on Methods and Models in Automation Robotics (MMAR)*, pp. 252–257. DOI: 10.1109/MMAR.2013.6669915 (cit. on p. 8).

Laporte, G., H. Mercure, and Y. Nobert (1987). “Generalized travelling salesman problem through n sets of nodes: the asymmetrical case”. In: *Discrete Applied Mathematics* 18.2, pp. 185–197. ISSN: 0166-218X. DOI: http://dx.doi.org/10.1016/0166-218X(87)90020-5 (cit. on p. 14).

LaValle, S. M. (2006). *Planning Algorithms*. New York, NY, USA: Cambridge University Press. ISBN: 0521862051 (cit. on pp. 17, 18, 20, 21).

LaValle, S. M. and S. A. Hutchinson (1998). “Optimal motion planning for multiple robots having independent goals”. In: *IEEE Transactions on Robotics and Automation* 14.6, pp. 912–925. DOI: 10.1109/70.736775 (cit. on p. 19).

LaValle, S. M. and J. J. Kuffner (2000). “Rapidly-Exploring Random Trees: Progress and Prospects”. In: *Algorithmic and Computational Robotics: New Directions*, pp. 293–308 (cit. on p. 21).

Lin, M. C., D. Manocha, and J. Cohen (1996). *Collision Detection: Algorithms and Applications* (cit. on p. 22).

Lin, S. and B. W. Kernighan (Apr. 1973). “An Effective Heuristic Algorithm for the Traveling-Salesman Problem”. In: 21.2, pp. 498–516. ISSN: 0030-364X. DOI: 10.1287/opre.21.2.498 (cit. on p. 14).

Nilsson, N. J. (1980). *Principles of Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN: 0-934613-10-9 (cit. on p. 21).

Noon, C. E. and J. C. Bean (1993). “An Efficient Transformation Of The Generalized Traveling Salesman Problem”. In: *INFOR: Information Systems and Operational*

*Research* 31.1, pp. 39–44. DOI: 10.1080/03155986.1993.11732212. eprint: http://dx.doi.org/10.1080/03155986.1993.11732212 (cit. on p. 15).

O'Donnell, P. A. and T. Lozano-Perez (1989). "Deadlock-free and collision-free coordination of two robot manipulators". In: *Proceedings, 1989 International Conference on Robotics and Automation*, 484–489 vol.1. DOI: 10.1109/ROBOT.1989.100033 (cit. on pp. 19, 22).

Olmi, R. and C. Secchi C.and Fantuzzi (Jan. 2011). "Coordination of industrial AGVs". In: *International Journal of Vehicle Autonomous Systems* 9, pp. 5–25. DOI: 10.1504/IJVAS.2011.038177 (cit. on p. 9).

Olmi, R., C. Secchi, and C. Fantuzzi (2008). "Coordination of multiple AGVs in an industrial application". In: *2008 IEEE International Conference on Robotics and Automation, ICRA 2008, May 19-23, 2008, Pasadena, California, USA*, pp. 1916–1921. DOI: 10.1109/ROBOT.2008.4543487 (cit. on p. 9).

Pallottino, L., E. Feron, and A. Bicchi (Mar. 2002). "Conflict Resolution Problems for Air Traffic Management Systems Solved with Mixed Integer Programming". In: *IEEE Transactions Intelligent Transportation Systems* 3.1, pp. 3–11 (cit. on pp. 9, 30).

Patriksson, M., A. Evgrafov, N. Andréasson, E. Gustavsson, and M. Önnheim (Oct. 2013). *An Introduction to Continuous Optimization*. ISBN: 9789144060774 (cit. on p. 23).

Pellegrinelli, S., N. Pedrocchi, L. Molinari Tosatti, A. Fischer, and T. Tolio (2017). "Multi-robot spot-welding cells for car-body assembly: Design and motion planning". In: *Robotics and Computer-Integrated Manufacturing* 44, pp. 97–116. ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2016.08.006 (cit. on p. 8).

Rambau, J. and C. Schwarz (2014). "Solving a vehicle routing problem with resource conflicts and makespan objective with an application in car body manufacturing". In: *Optimization Methods and Software* 29.2, pp. 353–375. DOI: 10.1080/10556788.2013.768993. eprint: https://doi.org/10.1080/10556788.2013.768993 (cit. on p. 8).

Riazi, S., C. Seatzu, O. Wigström, and B. Lennartson (2013). "Benders/gossip methods for heterogeneous multi-vehicle routing problems". In: *2013 IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA)*, pp. 1–6. DOI: 10.1109/ETFA.2013.6647983 (cit. on p. 16).

Saha, M. and P. Isto (Nov. 2006). "Multi-Robot Motion Planning by Incremental Coordination". In: pp. 5960–5963. DOI: 10.1109/IROS.2006.282536 (cit. on p. 18).

Salman, R., J. S. Carlson, F. Ekstedt, D. Spensieri, J. Torstensson, and R. Söderberg (2016). "An industrially validated CMM inspection process with sequence constraints". In: *Procedia CIRP* 44, pp. 138–143 (cit. on p. 14).

Segeborn, J., D. Segerdahl, J. S. Carlson, A. Carlsson, and R. Söderberg (2010). "Load balancing of welds in multi station sheet metal assembly lines". In: *ASME International Mechanical Engineering Congress and Exposition*. Vol. 44274, pp. 625–630 (cit. on pp. 3, 8).

Segeborn, J., D. Segerdahl, F. Ekstedt, J. S. Carlson, and M. Andersson (2014). “Industrially Validated Method for Weld Load Balancing in Multi Station Sheet Metal Assembly Lines”. In: *Journal of Manufacturing Science and Engineering* 136.1 (cit. on p. 3).

Skutella, M. and W. Welz (2011). “Route planning for robot systems”. In: *Operations research proceedings 2010*. Springer, pp. 307–312 (cit. on p. 8).

Spensieri, D., F. Ekstedt, J. Torstensson, R. Bohlin, and J. S. Carlson (2010). “Throughput Maximization by Balancing, Sequencing and Coordinating Motions of Operations In Multi-Robot Stations”. In: *Proceeding of the 8th International NordDesign Conference* (cit. on p. 40).

Toth, P. and D. Vigo (2002). *The Vehicle Routing Problem*. Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics. ISBN: 9780898715798 (cit. on p. 16).

Wigström, O., B. Lennartson, A. Vergnano, and C. Breitholtz (2013). “High-Level Scheduling of Energy Optimal Trajectories”. In: *IEEE Transactions on Automation Science and Engineering* 10.1, pp. 57–64. DOI: `10.1109/TASE.2012.2198816` (cit. on p. 3).

Wingquist Laboratory (Feb. 2017). URL: `http://www.chalmers.se/en/centres/wingquist/Pages/default.aspx` (cit. on pp. 3, 7).

Wingquist Laboratory VINN Excellence Centre (Feb. 2017). URL: `https://www.chalmers.se/en/centres/wqlvinnex/Pages/default.aspx` (cit. on p. 7).

Xin, J., C. Meng, F. Schulte, J. Peng, Y Liu, and R. R. Negenborn (2020). “A Time-Space Network Model for Collision-Free Routing of Planar Motions in a Multirobot Station”. In: *IEEE Transactions on Industrial Informatics* 16.10, pp. 6413–6422. DOI: `10.1109/TII.2020.2968099` (cit. on p. 8).