

JULIAN RICARDO DIAZ POSADA

Optimized Model-Based Path Generation for Robotic Manufacturing Processes



Universität Stuttgart



Fraunhofer
IPA

STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG BAND 106

Herausgeber:

Univ.-Prof. Dr.-Ing. Thomas Bauernhansl

Univ.-Prof. Dr.-Ing. Kai Peter Birke

Univ.-Prof. Dr.-Ing. Marco Huber

Univ.-Prof. Dr.-Ing. Dipl.-Kfm. Alexander Sauer

Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl

Univ.-Prof. a.D. Dr.-Ing. Prof. E.h. Dr.-Ing. E.h. Dr. h.c. mult. Engelbert Westkämper

Julian Ricardo Diaz Posada

**Optimized Model-Based Path Generation
for Robotic Manufacturing Processes**

FRAUNHOFER VERLAG

Kontaktadresse:

Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart
Nobelstraße 12, 70569 Stuttgart
Telefon 07 11/9 70-11 01
info@ipa.fraunhofer.de; www.ipa.fraunhofer.de

STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG

Herausgeber:

Univ.-Prof. Dr.-Ing. Thomas Bauernhansl^{1,2}

Univ.-Prof. Dr.-Ing. Kai Peter Birke^{1,4}

Univ.-Prof. Dr.-Ing. Marco Huber^{1,2}

Univ.-Prof. Dr.-Ing. Oliver Riedel³

Univ.-Prof. Dr.-Ing. Dipl.-Kfm. Alexander Sauer^{1,5}

Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl³

Univ.-Prof. a. D. Dr.-Ing. Prof. E.h. Dr.-Ing. E.h. Dr. h.c. mult. Engelbert Westkämper^{1,2}

¹ Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart

² Institut für Industrielle Fertigung und Fabrikbetrieb (IFF) der Universität Stuttgart

³ Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW)
der Universität Stuttgart

⁴ Institut für Photovoltaik (IPV) der Universität Stuttgart

⁵ Institut für Energieeffizienz in der Produktion (EEP) der Universität Stuttgart

Titelbild: © Fraunhofer IPA; Rainer Bez

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Daten sind im Internet über www.dnb.de abrufbar.

ISSN: 2195-2892

ISBN (Print): 978-3-8396-1614-7

D 93

Zugl.: Stuttgart, Univ., Diss., 2019

Druck: Mediendienstleistungen des Fraunhofer-Informationszentrum Raum und Bau IRB, Stuttgart
Für den Druck des Buches wurde chlor- und säurefreies Papier verwendet.

© by **FRAUNHOFER VERLAG**, 2020

Fraunhofer-Informationszentrum Raum und Bau IRB

Postfach 80 04 69, 70504 Stuttgart

Nobelstraße 12, 70569 Stuttgart

Telefon 07 11 9 70-25 00

Telefax 07 11 9 70-25 08

E-Mail verlag@fraunhofer.de

URL <http://verlag.fraunhofer.de>

Alle Rechte vorbehalten

Dieses Werk ist einschließlich aller seiner Teile urheberrechtlich geschützt. Jede Verwertung, die über die engen Grenzen des Urheberrechtsgesetzes hinausgeht, ist ohne schriftliche Zustimmung des Verlages unzulässig und strafbar. Dies gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Speicherung in elektronischen Systemen.

Die Wiedergabe von Warenbezeichnungen und Handelsnamen in diesem Buch berechtigt nicht zu der Annahme, dass solche Bezeichnungen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und deshalb von jedermann benutzt werden dürften. Soweit in diesem Werk direkt oder indirekt auf Gesetze, Vorschriften oder Richtlinien (z.B. DIN, VDI) Bezug genommen oder aus ihnen zitiert worden ist, kann der Verlag keine Gewähr für Richtigkeit, Vollständigkeit oder Aktualität übernehmen.

Optimized Model-Based Path Generation for Robotic Manufacturing Processes

Von der Fakultät Konstruktions-, Produktions- und
Fahrzeugtechnik der Universität Stuttgart
zur Erlangung der Würde eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Abhandlung

Vorgelegt von
Julian Ricardo Diaz Posada
aus Bogotá D.C., Kolumbien

Hauptberichter: Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl
Mitberichter: Prof. Dr.-Ing. habil. Björn Hein

Tag der mündlichen Prüfung: 09.12.2019

Institut für Steuerungstechnik der Werkzeugmaschinen und
Fertigungseinrichtungen (ISW) der Universität Stuttgart

2020

Foreword of the author

First of all I want to thank God for giving me the opportunity to develop this enriching doctoral research. This thesis is a result of my work as a Research Fellow at the Fraunhofer Institute for Manufacturing Engineering and Automation (IPA) in Stuttgart, Germany. I would like to give special thanks to Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl and Prof. Dr.-Ing. habil. Björn Hein for supervising, supporting, and reviewing this research work. Particular, I am sincerely thankful to Dr. Thomas Dietz who as team leader guided me in the definition of the topic and who enriched this research with fruitful discussions. Also, I want to thank Junior-Prof. Dr.-Ing. Andreas Pott for giving me his feedback in different stages of this research work.

I am also very grateful with Dr. Ulrich Schneider with whom I started my career at IPA during my Master thesis and who supported me in the development of this thesis. My gratitude goes also to my colleagues Dipl.-Ing. Alexander Kuss, Dr. Milad Geravand, and Dr. Manuel Drust with whom I discussed the topic and who provided me with valuable feedback. I want to thank also all my colleagues within the department of Robot and Assistive Systems, especially to the members of the groups "Robot systems for production processes" and "Industrial robotics: handling and intralogistics". Special thanks to Christoph Martin M.Sc., Manuel Mönnig M.Sc., Dr. Susanne Oberer-Treitz, Gesine Schleth M.Sc., Dr. Valentin Schmidt, Dr. Alexander Spiller, and Patrick Stelzer M.Sc. with whom I had the opportunity to work in research and industrial projects. Further thanks to Dr. Maximo A. Roa who also provided me feedback in a stage of this work. My gratitude goes also to all the students who during my time at IPA supported my activities as research fellow and who were always motivated with the research and development topics performed at IPA.

I also want to acknowledge the support from Dipl.-Ing. Martin Hägele for his work as Head of Department, and from Dr. Werner Kraus who for some time lead the group in which I worked. Special thanks also to Dr. Thomas Dietz, Dr. Gauthier Hentz, and Martin Henkel M.Sc. for their feedback on the document. I want to acknowledge the proofreading of M.A. Luzia Schumacher who not just proofread this document but also all my publications. My gratitude also goes to Heide Kreuzburg who supported all the organizational issues related with this PhD.

To my wife Tzvetelina Hadjipetkova my most sincere acknowledgment and love for all her support and company during the years developing this thesis. I also want to thank all my family and relatives who during all my life supported me and who provided me with the education and taught me all the values which led me to reach this stage in my live. Thank you all for all the love, happiness, and advice given to me throughout all my life and during the development of this research work. To my daughter Isabella all my love. I am so happy that you are with us now. May God bless and provide you with good health, long life, and happiness.

Short Summary

Optimized, efficient, and intuitive robotic programming is still a challenge in manufacturing and one of the main reasons why robots are not widely implemented in small and medium-sized enterprises and not broadly used for tasks such as milling, deburring, or welding. Reasons for this challenge are the time-consuming, complex, primarily manually optimized, and expert-dependent path generation for robotic manufacturing processes.

In order to effectively and efficiently respond to the current product variability demands, small and medium-sized enterprises require easy and optimized programmable robotic manufacturing systems for achieving profitable and rapid changeover. To make up for this deficiency, this dissertation raises the following research question: How to automatically generate offline optimized paths in reaction to infeasibilities for robotic manufacturing processes under constraints and process optimization criteria? This research question is answered with the following proposed and evaluated thesis: Robotic manufacturing process paths can be automatically optimized under defined process criteria by methodologically configuring and deploying sample-based generation algorithms, based on the procedural interpretation of the product, process, and resource model-based components of the robotic manufacturing process.

The basis for defining this thesis is the synergy found between four different research areas on robotics, which constitutes the state of the art of this document. These areas are: (1) the offline programming for robotic manufacturing processes, (2) the sample-based path generation, (3) the knowledge representation in robotic manufacturing and (4) the research and advances on specific robotic manufacturing processes such as welding, milling, and deburring. In this research, current advancements on these different areas are taken into account in order to develop, implement, and test the proposed architecture and methodology. The state-of-the-art analysis underscores and emphasizes that the lack of knowledge formalization, structured approach, and model-based architectures used in synergy with state-of-the-art algorithms is the reason for not having automatic optimized programming under constraints and optimization criteria for robotic manufacturing processes; resulting in the main contribution of this work.

A novel approach is introduced for interpreting methodologically the components of a robotic manufacturing process using the product, process, and resource classification as input. Mathematical equations for the product features, selected by the end-user over the computer-aided design files of the workpiece in a simulation environment, are automatically generated based on models of each individual product feature. Likewise, the manufacturing process parameters are semantically described and used in order to relate the process and the resources, which are articulated industrial robots for the use-cases studied in this dissertation.

The approach interprets the product, process, and resource components for simplifying the path planning problem. This is achieved by parameterizing the translational product constraints and the optimal or reference rotations of the product features in one dimension and configuring the

process degrees of freedoms into a novel robotic manufacturing processes configuration space. This configuration space is introduced with the goal to relate product constraints with the degrees of freedom and constraints of the manufacturing process and to simplify the complexity of the problem by allowing its solution in reasonable computation times for industrial and practical applications as demonstrated in the implemented and evaluated use-cases.

Three different use-cases are presented for simulating and evaluating the optimal path planning architecture and approach. The purpose of the evaluation in three different use-cases is to demonstrate the configurability of the approach and the re-usability of the defined models and implemented functions among different robotic manufacturing processes and criteria. Optimized robotic paths are generated for the manufacturing processes in reaction to infeasible robot configurations such as collision, unreachabilities, or maximum joint limits. The first use-case explores automatic and optimized collision avoidance in robotic welding. The second use-case explores the optimization of robot stiffness during robotic milling by using the intrinsic degree of freedom of the milling process. The third use-case demonstrates the use of a two dimensional laser scanner for sensor-guided robotic deburring which is optimized in order to get favorable measurements while assuring the deburring constraints.

The simulation and validation of these three different robotic manufacturing use-cases validate the configurability of the architecture and approach. Furthermore, it is demonstrated that optimized paths for manufacturing processes can be automatically computed by using the novel approach and architecture. The approach introduced in this thesis show how the state-of-the-art development in robotic offline programming software, sample-based motion planning, robotic knowledge description, and modeling of specific robotic manufacturing processes can be used as input for improving robotic motions after further modeling the robotic manufacturing process and for generating automatic robot programs by using the proposed concept named Automatic Optimized Offline Programming (AOLP).

This contribution demonstrates that the required robot programming time and the final robot process quality is improved by using the proposed approach and architecture in the implemented use-cases. Comparison between the widely used teach-in approach and a commercial offline programming software demonstrates this improvement. This approach verifies that the programming of robots does not require experts if robotic manufacturing models are used; that the robotic manufacturing processes can be automatically optimized, using sample-based algorithms after the methodological interpretation of the robotic manufacturing process; and that the general architecture can be used for several use-cases which reflect the solution to the above-defined research problem. This work concludes with an outlook on further potentials of the optimized path planning for robotic manufacturing.

Zusammenfassung

Optimierte, effiziente und intuitive Roboterprogrammierung ist nach wie vor eine Herausforderung in der Roboterfertigung und einer der Hauptgründe dafür, dass Roboter in kleinen und mittleren Unternehmen (KMU) nicht weit verbreitet sind. Aufgaben wie Fräsen, Entgraten oder Schweißen können nicht allzu häufig mit Industrieroboter in KMUs eingesetzt werden, vor allem wegen der zeitaufwändigen, komplexen, vorwiegend manuell optimierten und von Experten abhängigen Bahnplanung für Roboterfertigungsprozesse (RMPs von seiner Abkürzung im Englischen „robotic manufacturing processes“).

Um effektiv und effizient auf die aktuellen Produktvariabilitätsanforderungen reagieren zu können, benötigen KMUs einfache und optimierte programmierbare Roboterfertigungssysteme, damit eine schnelle Umstellung lukrativ ist. Um diesen Mangel auszugleichen, wirft diese Dissertation folgende Forschungsfrage auf: Wie generiert man automatisch offline optimierte Bahnen für robotergestützte Fertigungsprozesse unter Einschränkungen und Prozessoptimierungskriterien? Diese Forschungsfrage wird mit der folgenden Behauptung beantwortet: Die Roboterbahnen für Fertigungsprozesse können unter definierten Prozesskriterien automatisch optimiert werden, indem probabilistische Algorithmen methodisch konfiguriert und eingesetzt werden. Diese Konfiguration basiert sich auf der „Produkt, Prozess und Ressource“ (PPR) modell-basierten Beschreibung und der verfahrenstechnischen Interpretation des RMPs.

Nach der Einführung des Themas in Kapitel 1 werden die Grundlagen und der Stand der Technik in Kapitel 2 vorgestellt. Grundlage für die Definition dieser Arbeit ist die Synergie, die zwischen vier verschiedenen aktuellen Forschungsbereichen, die in dem Stand der Technik ausführlich beschrieben werden, besteht. Diese Bereiche sind: (1) die Offline-Programmierung (OLP) für Robotersysteme, (2) die probabilistische Bahnplanung, (3) die Wissensrepräsentation in der Roboterfertigung und (4) die Forschung und Fortschritte bei bestimmten Roboterherstellungsprozessen, wie z.B. Schweißen, Fräsen und Entgraten. In dieser Forschungsarbeit werden aktuelle Fortschritte auf diesen verschiedenen Gebieten dargestellt, um die vorgeschlagene Architektur und Methodik zu entwickeln, zu implementieren und zu testen. Der Befund von dem Stand der Technik betont, dass mangelnde Wissensformalisierung, strukturierte Herangehensweise und modellbasierte Architekturen, die in Synergie mit modernen Algorithmen verwendet werden, der Grund dafür sind, dass es keine automatisch optimierte Programmierung unter Einschränkungen und Optimierungskriterien für RMPs gibt, was zu dem Hauptbeitrag dieser Arbeit führt.

Kapitel 3 stellt den verallgemeinerten Ansatz und die implementierte Architektur vor. Dieser Ansatz wird als „Automatic Optimized Offline Programming“ (AOLP) für Roboterfertigungsprozesse bezeichnet, aufgrund der Automatisierung im Vergleich zu dem regulären OLP. Der neuartige Ansatz wird zur methodischen Interpretation der Komponenten eines Roboterherstellungsprozesses unter Verwendung der PPR-Klassifizierung als Eingabe eingeführt. Mathematische Gleichungen für die Produktmerkmale,

die vom Endbenutzer über die Computer-Aided Design (CAD) -Dateien des Werkstücks in einer Simulationsumgebung ausgewählt werden, werden automatisch basierend auf Modellen jedes einzelnen Produktmerkmals beschreiben. Ebenso werden die Parameter des Fertigungsprozesses semantisch beschrieben und verwendet, um auch den Prozess zu verknüpfen. Die in dieser Dissertation untersuchten Ressourcen sind 6-Achs-Gelenkarmroboter.

Der Ansatz interpretiert die PPR-Komponenten, um das Bahnplanungsproblem zu vereinfachen. Dies wird erreicht, indem die translatorischen Produktbedingungen und die optimalen oder gewünschten Rotationen des Produkts in einer Dimension parametrisiert werden. Die Prozessfreiheitsgrade werden ebenfalls in einem neuartigen Konfigurationsraum „C-space“ mit dem Namen RMP C-space konfiguriert. Dieser C-Raum wird eingeführt mit dem Ziel, Produktbedingungen mit den Freiheitsgraden und Bedingungen des Herstellungsprozesses in Beziehung zu setzen und die Komplexität des Problems zu vereinfachen. Dadurch wird die Bahnplanung in diesem C-space in vernünftigen Berechnungszeiten für industrielle und praktische Anwendungen ermöglicht, wie in den evaluierten Anwendungsfällen gezeigt wird.

In den Kapiteln 4 bis 6 werden die Implementierung der Anwendungsfälle, die experimentelle Untersuchung und die Ergebnisse dargestellt. Es werden drei verschiedene Anwendungsfälle zur Simulation und experimentellen Auswertung der optimalen Bahnplanungsarchitektur und des Ansatzes vorgestellt. Der Zweck der Bewertung in drei verschiedenen Anwendungsfällen besteht darin, die Konfigurierbarkeit des Ansatzes und die Wiederverwendbarkeit der definierten Modelle und implementierten Funktionen zwischen verschiedenen Fertigungsprozessen und -kriterien für Roboter zu demonstrieren. Optimierte Roboterbahnen werden für die Fertigungsprozesse als Reaktion auf nicht realisierbare Roboterkonfigurationen wie Kollisionen, Unerreichbarkeiten oder Überschreitungen der Gelenkgrenzen generiert.

Kapitel 4 stellt den ersten Anwendungsfall vor, die automatische und optimierte Kollisionsvermeidung beim Roboterschweißen. Die Vermeidung von Kollisionen ist eine zeitaufwändige Aufgabe, selbst bei den modernsten Methoden wie Teach-In und die OLP-Ansätze. Dies begründet die Implementierung des vorgeschlagenen Ansatzes. Die herstellungsoptimale Funktion wurde als Bewertung des minimal erforderlichen Winkels einer nicht kollisionserzeugenden Roboterkonfiguration zum Erreichen der optimalen Ausrichtung für das Schweißen definiert. Die Simulation wurde für zwei unterschiedliche Werkstücke mit kontinuierlichen Merkmalen vorgestellt, das erste mit linearen und das zweite aus einem halbumlaufenden und einem linearen Merkmal, um die Modularität des Ansatzes zu zeigen.

Experimente für den ersten Anwendungsfall wurden durch Vergleich des vorgeschlagenen AOLP und des Teach-In-Ansatzes durchgeführt. Die Programmierzeit mit dem AOLP ist 4.23 mal schneller als der Teach-In-Ansatz für die gegebene Schweißaufgabe. Darüber hinaus verbesserte das AOLP die Homogenität der geschweißten Nahtdicke um 35.62 %. Der Ansatz wurde auch mit einer kommerziellen OLP-Software verglichen, um eine spezifische Kollisionsvermeidung während des Roboterschweißens zu erreichen. Der Vergleich zeigte auch, dass das AOLP schneller ist als die kommerzielle OLP-Software (3.84 mal schneller). Als Erweiterung des Ansatzes wurde die nahezu optimale Werkstückpositionierung für das Roboterschweißen, um Gelenkbewegungen zu minimieren, vorgestellt und validiert.

Kapitel 5 beschreibt den zweiten Anwendungsfall. In diesem Anwendungsfall wird die Optimierung der Robotersteifigkeit beim Roboterfräsen anhand des intrinsischen Freiheitsgrades des Fräsprozesses untersucht. Die Optimierung wurde definiert als die Minimierung von Translationsabweichungen als Folge

der Roboternachgiebigkeit, die durch die in jedem Gelenk angebrachten Drehmomente verursacht wird, welche auf Prozesskräfte und Gelenkgewichte zurückzuführen sind. Das simulierte und gefräste Produktmodell besteht aus vier linearen Merkmalen, die ein Quadrat bilden. Es wurden zwei unterschiedliche Strategien zur Interpretation des Produkts und zur Definition des Bahnplanungsproblems adressiert. Der Fokus auf diese beiden Strategien wurde in diesem Anwendungsfall aufgrund der Diskontinuitäten in der „Costmap“ gegeben. Die Diskontinuitäten sind Resultat der Optimierungsfunktion in Bezug auf die Produktmerkmalsdefinitionen. Beim Roboterfräsen wurden die optimierten Bahnen mit den nicht optimierten Bahnen verglichen. Nach der Messung der Frästiefen mit einer Koordinatenmessmaschine (KMG) werden für beide Strategien Verbesserungen der mittleren absoluten Abweichung der Frästiefen aufgezeichnet (51.17 % und 57.40 % mit der „single“ und „multiple“ Strategien beziehungsweise).

Kapitel 6 erörtert den dritten Anwendungsfall und demonstriert die Verwendung eines zweidimensionalen Laserscanners zum sensorgesteuerten Entgraten von Robotern, der optimiert ist, um günstige Messungen zu erhalten und gleichzeitig die Entgratungsbedingungen zu gewährleisten. Die manuelle Programmierung dieses Anwendungsfalls ist zeitaufwändig, da die Entgratungsbedingungen beim Drehen des „Tool Center Point“ (TCP) zur Erzielung optimierter oder gültiger Messungen mit dem Sensor kompliziert sind. Die zu optimierende Funktion wurde anhand zwei Kriterien definiert. Symmetriekosten wurden festgelegt, um den maximalen Informationsgehalt aus den gemessenen Merkmalen zu erhalten, und es wurden Tiefenkosten eingeführt, um den Ort festzulegen, an dem das 2D-Profil gemessen wird. Die Roboterkonfiguration ist als durchführbar definiert, wenn das Sichtfeld des Sensors das Werkstück für eine Messung schneidet. Darüber hinaus wurde eine Zustandsmaschine implementiert, um die verschiedenen Schritte des sensorgeführten Prozesses zu berücksichtigen. Die Optimierung wurde für ein Produkt mit fünf linearen kontinuierlichen Merkmalen unter Verwendung eines optimalen probabilistischen Algorithmus simuliert. Die Simulation wurde unter Verwendung des vorgeschlagenen Ansatzes und der Architektur durchgeführt und analysiert.

Die Simulation und Validierung dieser drei verschiedenen Anwendungsfälle für die Roboterfertigung bestätigen die Konfigurierbarkeit der Architektur und des Ansatzes. Des Weiteren wird gezeigt, dass optimierte Bahnen für Fertigungsprozesse mithilfe des neuartigen Ansatzes und der neuen Architektur automatisch berechnet werden können. Der vorgestellte Ansatz zeigt auch, wie die Weiterentwicklung in der robotergestützten Offline-Programmierung (OLP), der probabilistischen Bahnplanung, der Wissensbeschreibung in der Robotik und der Modellierung spezifischer Fertigungsprozesse von Robotern in dem AOLP-Ansatz einbezogen werden kann.

Diese Arbeit endet und präsentiert weitere Perspektiven der optimierten Bahnplanung für die Roboterfertigung in Kapitel 7. Der wissenschaftliche Beitrag zeigt anhand der Anwendungsbeispiele, dass die erforderliche Roboterprogrammierungszeit und die endgültige Qualität des Roboterprozesses durch die Verwendung des vorgeschlagenen Ansatzes und der vorgeschlagenen Architektur verbessert werden können. Der Vergleich zwischen dem weit verbreiteten Teach-In-Ansatz und kommerzieller OLP-Software beweist diese Verbesserung. Dieser Ansatz stellt sicher, dass für die Programmierung von Robotern keine Experten erforderlich sind, wenn Roboter-Fertigungsmodelle verwendet werden. Außerdem können die Roboterherstellungsprozesse nach der methodologischen Interpretation des RMP automatisch mit probabilistischen Algorithmen optimiert werden. Die Optimierung basiert sich auf eine allgemeine Architektur, die für mehrere Anwendungsfälle verwendet werden kann, die die Lösung des oben definierten Forschungsproblems widerspiegeln.

Table of Contents

Foreword of the author	iii
Short Summary	iv
List of Figures	xv
List of Tables	xvii
List of Abbreviations	xviii
List of Symbols	xx
1 Introduction	1
1.1 Motivation	1
1.2 Problem Statement	2
1.3 Thesis Statement and Objectives	4
1.3.1 Focus of the Thesis and its Delimitation	5
1.4 Structure of this Thesis	6
2 Fundamentals and State of the Art	7
2.1 Path Generation for Robotic Manufacturing Processes	7
2.1.1 Programming Methods	8
2.1.2 Comparison among Programming Methods	11
2.1.3 Path Generation Algorithms	14
2.1.3.1 Grid-based Search	15
2.1.3.2 Mathematical Computation Algorithms	15
2.1.3.3 Bio-inspired Algorithms	16
2.1.3.4 Sample-Based Algorithms	16
2.1.4 Path Generation Libraries	22
2.1.5 Comparison among Path Generation Algorithms	22
2.2 Knowledge Representation in Robotic Manufacturing	25
2.2.1 The Product, Process, and Resource Model	27
2.2.2 The Product Model	28
2.2.3 The Process Model	29
2.2.4 The Resource/Robot Model	29
2.3 Offline Programming Software for Robotic Manufacturing Processes	30
2.4 Path Generation in Robotic Welding	33
2.4.1 Methods and Algorithms	33
2.4.2 Knowledge-Based Approach and Offline Programming	34

2.4.3	Collision Avoidance	34
2.4.4	Path Optimization	35
2.5	Path Generation for Improving Robotic Milling Accuracy	36
2.5.1	Error Compensation	38
2.5.2	Robotic Compliance Models	39
2.5.3	Path Optimization	40
2.6	Path Generation for Error Compensation in Robotic Deburring	42
2.6.1	Offline Compensation and Programming	42
2.6.2	Online Compensation and Programming	43
2.7	Proposition	43
2.7.1	Synergy Among State-of-the-Art Research Areas	44
2.7.2	Problematic and Motivation in the Studied Robotic Manufacturing Processes	46
3	The Automatic Optimized Offline Programming Approach for Robotic Manufacturing	49
3.1	Proposed Model-based Description of the Robotic Manufacturing Process	49
3.1.1	Product Model	51
3.1.1.1	Single product interpretation and problem strategy	54
3.1.1.2	Multiple product interpretation and problem strategy	55
3.1.2	Process Model	55
3.1.2.1	Geometrical Definition	55
3.1.2.2	State and Motion Cost	56
3.1.3	Robot Model	60
3.2	Robotic Manufacturing Process Configuration Space ($\mathbb{RMP}^n$)	61
3.2.1	Feasible Region of the Robotic Manufacturing Process Configuration Space	63
3.3	Formal Problem Formulation	65
3.4	Proposed Approach: The Automatic Optimized Offline Programming	66
3.5	Approach Extension: The Automatic Close-to-Optimal Workpiece Positioning	74
3.6	Architecture and Software Implementation	75
3.7	Conclusion	78
4	Optimized Collision Avoidance in Robotic Welding	80
4.1	Problem Definition	80
4.2	Product, Process, and Resource Modeling	80
4.2.1	Product Model	80
4.2.2	Process Model	83
4.2.3	Robot Model	84
4.3	State and Motion Cost	84
4.3.1	State Cost	84
4.3.2	Motion Cost	85
4.4	Optimized Path Generation for Collision Avoidance in Robotic Welding	85
4.5	Optimization in OLP Software and Experimentation	87
4.5.1	Experimental Setup	87
4.5.2	Optimization in OLP Software	87
4.5.3	Welding Robot Experimentation	100
4.5.4	Close-to-Optimal Workpiece Positioning for Robotic Welding	108

4.6	Conclusion	114
5	Optimized Stiffness for Robotic Milling	115
5.1	Problem Definition	115
5.2	Product, Process, and Resource Modeling	115
5.2.1	Product Model	115
5.2.2	Process Model	116
5.2.3	Robot Model: Wrench Computation due to Process Force and Joint Weights .	118
5.2.4	Robot Model: Joint Compliance Model	119
5.3	State and Motion Cost	120
5.3.1	State Cost	120
5.3.2	Motion Cost	122
5.4	Optimized Path Generation with Stiffness Optimization	122
5.5	Optimization in OLP Software and Experimentation	125
5.5.1	Experimental Setup	125
5.5.2	Optimization in OLP Software	126
5.5.3	Experimentation	131
5.5.4	Close-to-Optimal Workpiece Positioning for Robotic Milling	135
5.6	Conclusion	138
6	Optimized Sensing Strategy for Robotic Deburring	139
6.1	Problem Definition	139
6.2	Product, Process, and Resource Modeling	140
6.2.1	Product and Process Models	140
6.2.2	Robot and Sensor models	141
6.3	State and Motion Cost	142
6.3.1	State Cost	142
6.3.2	Motion Cost	143
6.4	Optimized Path Generation with Sensing Strategy Optimization	143
6.5	Optimization in OLP Software	144
6.5.1	Experimental Setup	144
6.5.2	Optimization in OLP Software	145
6.6	Conclusion	148
7	Conclusions and Outlook	149
7.1	Conclusions	149
7.2	Outlook	151
	Appendixes	153
A.	Robot Joint Mass and Center of Mass Estimation	153
B.	Parameters of the Joint Compliance Model for the Robot Kuka KR270 R2700 Ultra .	154
C.	Robot Tool Calibration and Accuracy	155
	Bibliography	156

List of Figures

1.1	Comparison of phases of robot use in mass production and SME production scenarios.	1
2.1	Robot programming methods.	8
2.2	Configuration space with feasible and infeasible regions, start and goal states, and a planned path.	18
2.3	The Product, Process, and Resource (PPR) Model.	28
2.4	PPR model in Delmia for the simulation of a robot task.	28
2.5	Robot stiffness pose optimization.	41
2.6	Synergy among knowledge areas of robotic manufacturing as potential for the automatic optimized path generation.	44
3.1	Approach and architecture inputs: model-based description of the robotic manufacturing process	50
3.2	World, product, and process coordinate systems and their transformations.	50
3.3	Product models of translational constraints for different features.	53
3.4	Illustrations of the different product interpretations.	53
3.5	Illustration of the process model and the product to process transformation.	56
3.6	Illustration of the state cost.	57
3.7	Ratio cost factor versus the normalized angular ratio in order to constrain the angular velocity.	59
3.8	Configuration spaces involved in the optimal path generation.	62
3.9	Example of the robotic manufacturing process configuration space and the mapping of samples to the Cartesian space.	63
3.10	Robot lengths and transformations involved in the heuristic for the reachability computation.	65
3.11	Inputs required by the general steps of the approach.	67
3.12	Workflow of the approach.	68
3.13	Representation of a robotic manufacturing process configuration space with extra dimension depicting the state cost.	70
3.14	Flow diagram for the close-to-optimal workpiece positioning.	75
3.15	General architecture of the automatic optimal path generation for robotic manufacturing processes.	76
3.16	Model-based description in AML using the PPR model categorization.	77
3.17	Overview of the software implementation.	79
4.1	Product models of workpieces for the welding use-case.	81

4.2	Illustration of two continuous T-joint seam product models and their notations.	82
4.3	Welding robot cell.	87
4.4	Collision avoidance in $\mathbb{RMP}^2$, joint movements, and robot sequences using a process degree of freedom for avoiding automatically a collision.	88
4.5	Evaluated valid states and optimized interconnected path in the $\mathbb{RMP}^2$ for avoiding a collision using the rotation around the welding rotation vector.	89
4.6	Optimized robot collision avoidance over two continuous linear features.	90
4.7	Real and simulated workpiece with a semi-circumferential and a linear feature and the collision objects	91
4.8	Transformation from the product (p) to the process ($proc$) coordinate system for the welding use-case.	92
4.9	State cost elevation map, $\mathbb{RMP}$ samples, and computed path for the semi-circumferential and linear product features.	93
4.10	Total motion cost evaluations and fitted function for the automatic termination of the RRT*.	93
4.11	Heuristic evaluations for the automatic termination of the RRT* algorithm for the semi-circumferential feature.	94
4.12	Heuristic evaluations for the automatic termination of the RRT* algorithm for the linear feature.	94
4.13	Path planning over the welding use-case $\mathbb{RMP}^2$ constraining the angular velocity with respect to k with different b_v factors for the collision avoidance through the semi-circumferential feature.	95
4.14	3D representation of the planned path in the $\mathbb{RMP}^3$ for the optimized collision avoidance of a semi-circumferential feature using two process DoFs.	97
4.15	2D-projections of the planned path in the $\mathbb{RMP}^3$ using two process DoFs.	97
4.16	3D views depicting the collision and the planned path in the $\mathbb{RMP}^3$ using two product DoFs.	97
4.17	Comparison of the final total motion cost with different optimal sample-based algorithms for the collision avoidance of the semi-circumferential feature.	98
4.18	Comparison of the required plannings durations with optimal sample-based algorithms for the collision avoidance of the semi-circumferential feature.	99
4.19	Different path planning algorithms depicting their vertices and computed paths. . . .	99
4.20	Sequence of robot movements in simulation and with the real robot for the optimized collision avoidance for welding a workpiece with continuous semi-circumferential and linear features.	101
4.21	Required programming time for the welding task involving collision avoidance using the AOLP and the teach-in approaches for all participants.	102
4.22	Influence of the programming experience on the required programming time for the welding task involving collision avoidance with the AOLP and the teach-in approaches for the groups of participants.	102
4.23	Laser scanner with the twenty measured throat thicknesses after overlaying images of each taken profile.	104

4.24	Absolute errors with respect to the mean of the throat thicknesses	104
4.25	Example of welded workpieces using the Automatic Optimized Offline Programming (AOLP) and the teach-in approach.	105
4.26	Simulation of the collision avoidance in two different OLP environments.	107
4.27	Simulation and real robotic welding cell.	108
4.28	Validity evaluation for the workpiece positioning in the welding use-case.	109
4.29	Costmaps for the workpiece positioning optimization in the welding use-case.	110
4.30	Robot joint movements for the optimized and worst welding workpiece position with joint movement optimization.	111
4.31	Weighted costmap for the workpiece positioning in the welding use-case with the simulated annealing algorithm and the computed close-to-optimal workpiece pose. . .	112
4.32	Path planning and costmaps for different rotatory table configurations.	113
5.1	Illustration of two milling paths on the product and its notations.	117
5.2	Joint compliance models for the Robot Kuka KR270 R2700 Ultra and their non-linear descriptions around the $\vec{z}$ axes of all six joints.	120
5.3	State cost computation for the stiffness optimization in robotic milling.	121
5.4	Real and simulated robotic milling cell.	125
5.5	Robot end-effector with the milling tool.	125
5.6	STL model of the real workpiece with 4 selected edges and its corners A, B, C, and D. .	126
5.7	Sequence of robot movements for a four-edge machining task without optimization following the product convention.	127
5.8	State cost elevation map of the path generation problem for optimizing stiffness in robotic milling.	127
5.9	$\mathbb{RMP}^2$ for the four milled paths with their state cost (Euclidean robot translation deviations) and a computed optimized path using the T-RRT algorithm with single problem strategy.	128
5.10	Sequence of robot movements for a four-edge machining task with optimized stiffness after generating the path with the T-RRT algorithm using the single problem strategy. .	129
5.11	$\mathbb{RMP}^2$ for the four milled paths with their state cost (Euclidean robot translation deviations) with the computed optimized path using the T-RRT algorithm and the multiple problem strategy, the simplified path, and the path used to compare the milling accuracy.	130
5.12	Translational deviations on x , y , z for the found solution based on the models.	132
5.13	Machined workpieces with corners A, B, C, and D using the single problem strategy with and without optimization.	132
5.14	Depth deviations for the optimized and non-optimized mills with the single problem strategy.	133
5.15	Machined workpieces with corners A, B, C, and D using the multiple problem strategy with and without optimization.	134
5.16	Depth deviations for the optimized and non-optimized mills with the multiple problem strategy.	135

5.17	Simulation and real robotic milling cell.	136
5.18	Total motion cost for different workpiece positions with the results of two simulated annealings.	137
5.19	Path planning in the costmap for optimizing translation deviations due to robot compliance in the optimized workpiece position.	137
5.20	Path planning in the costmap for optimizing translation deviations due to robot compliance positioning the workpiece in the worst position.	137
6.1	Illustration of the sensor-guided process.	139
6.2	Illustration of the product model, the selected linear features, and its coordinates for the optimized sensing strategy use-case.	140
6.3	Deburring process state machine. (1) sensing, (2) deburring, (3) stop sensing, and (4) stop deburring.	141
6.4	Sensor measurement model for computing the expected measurement calculation. . . .	142
6.5	Illustration of the state cost for the deburring use-case.	142
6.6	Deburring robot cell.	144
6.7	Sensing state cost elevation map for the deburring use-case and final computed optimized path for the five product edges workpiece.	145
6.8	$\mathbb{RMP}^2$ for the five product edges with its sensing state costs and the computed optimized path using the RRT* algorithm for the proposed deburring use-case.	146
6.9	Sequence of robot motions for the sensor strategy optimization.	147
6.10	Computed optimized path in $\mathbb{SE}^3$ over the deburring workpiece.	147
A.1	Illustration of the center of mass for each joint of the Kuka KR270 R2700 ultra. . . .	153
C.1	Robot deviations due to rotation around the z-axis.	155

List of Tables

2.1	Comparison of programming methods in robotic manufacturing.	14
2.2	Comparison of programming algorithms in robotic manufacturing.	25
2.3	Some relevant OLP software for the generation of robotic paths for manufacturing processes including open-source packages.	31
4.1	Results and characteristics of the path planning for the semi-circumferential and linear features for the optimized collision avoidance in the robotic welding use-case.	92
4.2	Results and characteristics of the path planning constraining the angular velocity with respect to k for different ratio border values in the welding use-case.	96
4.3	Required path planning times in the robotic welding use-case using two process DoFs.	96
4.4	Results and characteristics of the sampling and optimization in the robotic welding use-case using two process DoFs.	96
4.5	Factors of the AOLP being faster than the teach-in approach.	103
4.6	Required programming time for the path planning for the semi-circumferential and linear features for all participants.	103
4.7	Mean of the absolute error with respect to the mean of the throat thickness for each participant and method.	105
4.8	Results and characteristics of the path planning for the semi-circumferential feature for the different participants.	106
4.9	Results and characteristics of the path planning for the linear feature for the different participants.	106
5.1	Required path planning times for the robotic milling use-case optimizing stiffness for the four linear features using the single problem strategy.	129
5.2	Sampling, optimization results, and characteristics in the robotic milling use-case optimizing stiffness for the four linear features using the single problem strategy.	129
5.3	Required Path planning times for the robotic milling use-case optimizing stiffness for the four linear features using the multiple problem strategy.	131
5.4	Sampling and optimization results and characteristics in the robotic milling use-case optimizing stiffness for the four linear features using the multiple problem strategy.	131
6.1	Times, results, and characteristics of the path planning for different maximum allowed sampling times using the RRT* algorithm for the sensor-guided deburring use-case.	147
A.1	Mass and center of mass of the end-effector and each joint for a Kuka KR270 R2700 ultra estimated based on its CAD model and its total weight.	154

B.1	Parameters for the linear functions and third order polynomials describing the joint compliance models for the robot Kuka KR270 R2700 ultra.	154
B.2	Limits for the linear functions of the compliance models for the robot Kuka KR270 R2700 ultra.	154
C.1	Robot deviations due to rotation around the z-axis.	155

List of Abbreviations

2D	two dimensional
3D	three dimensional
A*	A-star algorithm
AML	Automation Markup Language
AOLP	Automatic Optimized Offline Programming
BIT	Batch Informed Trees
C-space	Configuration Space
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
CMM	Coordinate-Measuring Machine
CNC	Computerized Numerical Control
Deg	Degrees
DH	Denavit-Hartenberg
DLL	Dynamic-Link Library
DoF	Degree of Freedom
FMT	Fast Marching Tree algorithm
GUI	Graphic User Interface
ICP	Iterative Closest Point
IPA	Fraunhofer Institute for Manufacturing Engineering and Automation
IR	Industrial Robot
ISO	International Organization for Standardization
MAG	Metal Active Gas
OLP	Offline Programming
OMPL	Open Motion Planning Library
OSBA	Optimal Sample-based Algorithm
PbD	Programming by Demonstration
PLC	Programmable Logic Controller
PPR	Product, Process, and Resource
PRM	Probabilistic Roadmap
RGB	Red Green Blue
RMP	Robotic Manufacturing Process
RRT	Rapidly-exploring Random Tree

s	Seconds
SME	Small and Medium-sized Enterprise
STEP	STandard for the Exchange of Product model data
STL	Stereolithography
T-RRT	Transition-based Rapid-Random Tree
TCP	Tool Center Point
w.r.t.	with respect to
XML	eXtensible Markup Language

List of Symbols

The following conventions and symbols are used throughout this dissertation.

Formatting

Example	Meaning
$\mathbb{R}$	C-space (blackboard bold)
a	Constant or variable (latin lower-case letter)
M	Matrix (capital letter)
$\mathbf{f}$	Mathematical function (bold)
$\mathcal{W}$	Set (calligraphic capital letter)
$\mathcal{x}$	Set element (calligraphic lower-case letter)
Func	Software function (roman)
$\underline{h}$	Tuple (underlined)
$\vec{p}$	Vector (upper arrow)

Symbols

Symbol	Description	Cross-reference	
C-space notations			
SE^3	Cartesian space	sec. 2.1.3.4	p. 16
$\mathbb{R}^n$	Euclidean space of n dimensions	sec. 2.1.3.4	p. 16
$\mathcal{W}$	Workspace	sec. 2.1.3.4	p. 16
$\mathcal{S}_{free}$	Feasible region	sec. 2.1.3.4	p. 16
$\mathcal{S}_{unfree}$	Infeasible region	sec. 2.1.3.4	p. 16
s	State, configuration, or sample	sec. 2.1.3.4	p. 16
s_s	Start state	sec. 2.1.3.4	p. 16
s_g	Goal state	sec. 2.1.3.4	p. 16
σ	Path	sec. 2.1.3.4	p. 16
σ^*	Optimized path	sec. 2.1.3.4	p. 18
$\mathcal{S}_s, \mathcal{S}_g$	Start and goal regions	sec. 3.1.1	p. 51
$\mathbb{J}^n$	Joint space of a robot of n dimensions	sec. 3.1.3	p. 60
RMP^n	Robotic manufacturing process C-space of n dimensions	sec. 3.2	p. 61

Symbol	Description	Cross-reference	
Constants or variables			
$n \in \mathbb{N}$	Number of dimensions of the RMP	sec. 2.1.3.4	p. 16
$k \in [k_s, k_g]$	Variable parameterizing the translation and desired or reference orientation of product features	sec. 3.1.1	p. 51
$k_s \in \mathbb{R}$	Initial value of k	sec. 3.1.1	p. 51
$k_g \in \mathbb{R}$	Final value of k	sec. 3.1.1	p. 51
$k_d \in \mathbb{R}$	End of parametrization of a single product feature	sec. 3.1.1	p. 51
$u \in \mathbb{N}$	Number of selected continues features	sec. 3.1.1	p. 51
$m \in \mathbb{N}$	Identifier for the selected features	sec. 3.1.1	p. 51
$l \in \mathbb{R}$	Length of a product feature	sec. 3.1.1	p. 51
$\varrho \in \mathbb{N}$	Number of path generation problems	sec. 3.1.1	p. 51
x, y, z	Translational components in $\mathbb{SE}^3$	sec. 3.1.2	p. 55
α, β, γ	Rotations around $\vec{x}$, $\vec{y}$, and $\vec{z}$ respectively	sec. 3.1.2	p. 55
$\delta\alpha, \delta\beta, \delta\gamma$	Rotational Degrees of freedom in α, β , and γ	sec. 3.1.2	p. 55
w_M	Weighting factor for cost functions with identifier M as example	sec. 3.1.2.2	p. 57
r_{max}	Maximum admissible angular velocity	sec. 3.1.2.2	p. 58
r_d	Desired angular velocity	sec. 3.1.2.2	p. 58
a_r	Angular ratio with respect to k	sec. 3.1.2.2	p. 58
b_v	Ratio border value	sec. 3.1.2.2	p. 58
r_c	Ratio cost factor	sec. 3.1.2.2	p. 58
j_{max}, j_{min}	Maximum and minimum joint values	sec. 3.2.1	p. 63
$d \in \mathbb{N}$	Number of cost functions	sec. 3.3	p. 65
ζ	Percentage of rotation between seams for the welding process	sec. 4.2.1	p. 80
m_j, m_{EE}	Joint mass and mass of the end-effector	sec. 5.2.3	p. 118
$s_a \dots s_h$	Parameters of the robot joint compliance models	sec. 5.2.4	p. 119
s_{max}, s_{min}	Lower and upper ranges describing the nonlinearities of the robot joint compliance models	sec. 5.2.4	p. 119
Coordinate frames and notations			
w	World frame	sec. 3.1	p. 49
p	Product frame	sec. 3.1	p. 49
$proc$	Process frame	sec. 3.1	p. 49
rb	Robot base frame	sec. 3.2.1	p. 64
tcp	Tool Center Point (TCP) frame	sec. 3.2.1	p. 64
wp	Workpiece frame	sec. 3.5	p. 74
cm_j	Center of mass frame of joint j	sec. 5.2.3	p. 118
sen	Sensor frame	sec. 6.2.1	p. 140

Symbol	Description	Cross-reference	
mea	Expected measurement of the workpiece frame	sec. 6.2.1	p. 140
Matrixes			
R	Rotation matrix	sec. 3.1	p. 49
T_a^b	Homogeneous transformation from a to b	sec. 3.1	p. 49
Software implementations of mathematical functions			
$C(\cdot)$	Cost function	sec. 2.1.3.4	p. 18
$R_q(\cdot)$	Function for calculating a rotation matrix from a quaternion	sec. 3.1.1	p. 51
$R_{eul}(\cdot)$	Function for calculating a rotation matrix from Euler angles	sec. 3.1.2	p. 55
$C_s(\cdot)$	State cost function	sec. 3.1.2.2	p. 56
$C_M(\cdot)$	Motion cost function	sec. 3.1.2.2	p. 56
$C_p(\cdot)$	Backward policy cost function	sec. 3.1.2.2	p. 56
$C_t(\cdot)$	Total motion cost function	sec. 3.1.2.2	p. 56
$FK(\cdot)$	Robot forward kinematics function	sec. 3.1.3	p. 60
$IK(\cdot)$	Robot inverse kinematics function	sec. 3.1.3	p. 60
$f(\cdot)$	Function for transforming from RMP^n to SE^3	sec. 3.2	p. 61
$C_{wp}(\cdot)$	Workpiece positioning cost function	sec. 3.5	p. 74
$Q2AA(\cdot)$	Function to calculate the axis-angle convention from a quaternion	sec. 4.2.1	p. 80
$R_{aa}(\cdot)$	Function for transforming from axis-angle into rotational matrix notation	sec. 4.2.1	p. 80
$f_{stiff}(\cdot)$	Robot joint stiffness model	sec. 5.1	p. 115
$f_{compl}(\cdot)$	Robot joint compliance model	sec. 5.2.4	p. 119
$Tw(\cdot)$	Force-moment transformation	sec. 5.2.3	p. 118
Tuples			
$\underline{q}$	Quaternions tuple	sec. 3.1.1	p. 51
$\underline{j}$	Robot joints configurations tuple	sec. 3.1.3	p. 60
$\underline{x}$	Cartesian positions and orientations tuple	sec. 3.1.3	p. 60
$\underline{B}$	Bounds for the RMP^n tuple	sec. 3.4	p. 66
$\underline{\omega}_A$	Wrench tuple in coordinate system A	sec. 5.2.3	p. 118
$\underline{\tau}_j$	Joints moment tuple	sec. 5.2.3	p. 118
$\underline{\delta x}$	Deviations in Cartesian space tuple	sec. 5.3.1	p. 120
$\underline{\delta j}$	Deviations in joint space tuple	sec. 5.3.1	p. 120

Symbol	Description	Cross-reference	
Vectors			
$\vec{V}$	Position vector	sec. 3.1.1	p. 51
$[\theta_m, \vec{T}_m]$	Angle (θ_m) and axis ($\vec{T}_m$) between two rotations calculated with the axis-angle convention	sec. 4.2.1	p. 80
$\vec{F}$	Force vector	sec. 5.2.2	p. 118
$\vec{F}_{Proc}$	Milling process force vector	sec. 5.2.2	p. 118
$\vec{\tau}$	Moments vector	sec. 5.2.3	p. 118
Other software functions			
IsValid($\cdot$)	Validity evaluation of a robot configuration	sec. 3.2.1	p. 63
Interpret($\cdot$)	Interpretation of the Product Process and Resource (PPR) components of the robotic manufacturing process	sec. 3.4	p. 66
RMPtoSE ³ ($\cdot$)	Computation of the transformation from $\mathbb{RMP}^n$ to $\mathbb{SE}^3$ ($\mathbf{f}$)	sec. 3.4	p. 66
StartGoalStates($\cdot$)	Definition of the start and goal states	sec. 3.4	p. 66
DetermineDim($\cdot$)	Determination of the dimensions of the $\mathbb{RMP}^n$	sec. 3.4	p. 66
Bounds($\cdot$)	Definition of the bounds of the $\mathbb{RMP}^n$	sec. 3.4	p. 66
Fit($\cdot$)	Fitting of a negative exponential function	sec. 3.4	p. 66
OSBA($\cdot$)	Optimal sample-based algorithm	sec. 3.4	p. 66
CollisionRange($\cdot$)	Determination of the range of the collision in the k dimension	sec. 4.4	p. 85
Semantic models			
P	Product model	sec. 3.1.1	p. 51
PROC	Process model	sec. 3.1.2	p. 55
ROBOT	Robot model	sec. 3.1.3	p. 60

1 Introduction

1.1 Motivation

Manufacturing production requirements and market demands have been evolving drastically during the last decades. Nowadays, mass production is not the only requirement, as product variability and customization combined with the growing amount of Small and Medium-sized Enterprises (SMEs) producing goods also play an important role in the current production scenario (Marvel, Messina et al. 2015, Dietz, Schneider et al. 2012, Nilsson, Johansson et al. 2005). Robotics and automation solutions in manufacturing have evolved and emerged in order to face the mass production. Furthermore, robots are considered flexible and universal machines, which can perform several tasks. Notwithstanding, these solutions still need to be further developed in order to satisfy the changeability and adaptability requirements that will allow for the wide adoption of Industrial Robots (IRs) for manufacturing by SMEs (Hägele, Nilsson et al. 2008).

Under current market constraints, robots cannot be used cost-effectively for a wide range of production tasks in SMEs as the production is driven by smaller lot sizes for several product variants, which demands frequent changeover (Dietz, Schneider et al. 2012). A crucial factor for satisfying the changeability and adaptability requirements of robotic systems for SMEs is an effective and efficient production configuration. To further illustrate this need, Figure 1.1 depicts a comparison of the phases of robot use between mass and SME production (Dietz, Schneider et al. 2012).

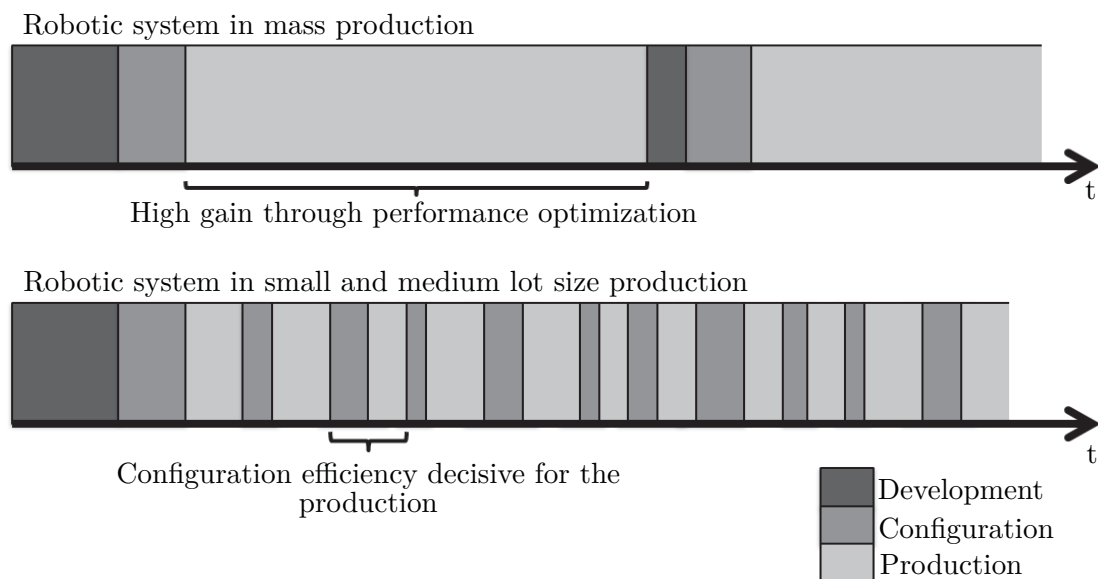


Figure 1.1: Comparison of phases of robot use in mass production and SME production scenarios (Dietz, Schneider et al. 2012).

An essential issue of robot configuration is its programming. The spectrum of practical manufacturing applications can be increased if robots were easier to program (Hägele, Nilsson et al. 2008). The reduction of the programming time and the improvement of the process quality achieved with industrial robots will enhance the benefit of these systems, pushing forward its investment and adoption by SMEs. The development of approaches and architectures facilitating end-users to program a robot with a higher level of abstraction is an intrinsic requirement of the current production scenario (Pires 2009). Therefore, with the motivations of improving the programming experience and the process quality in Robotic Manufacturing Processes (RMPs), the problem is stated below.

1.2 Problem Statement

Robotic systems for manufacturing purposes have become more common in the industry due to their versatility and low cost. However, in order for robots to be widely utilized by SMEs, current re-configuration and high-quality production demands to address time-consuming, expert-dependent, non-optimized, and complex programming processes.

Existing programming methods do not completely satisfy the current production demands. The primary methods used in industry, known as the teach-in and the Offline Programming (OLP) methods, are time-consuming and still present disadvantages for many industrial sectors. The teach-in approach requires stopping the robot during the programming, which affects the productivity of these systems. Moreover, if the geometries of the products are complex and the scenario requires dexterous robot movements, the teach-in process demands significant programming time because a programmer must configure many complex poses. Costly and rare experts that are capable of operating the robot controller and optimally setting the manufacturing process are required when using this method.

The OLP method does not require to interrupt manufacturing. However, to program the path in the simulation environment is also a time-consuming task, which requires qualified personnel. In case of infeasible robot movements with respect to the process (such as collisions, unreachabilities, or maximum joint limits), the end-user needs to operate the robot in the simulation manually in order to generate feasible paths. Using this method, the process criteria are rarely optimized. Moreover, optimized paths are only generated in stand alone implementations and not based on a general approach. Despite this, the OLP approach clusters several knowledge representation of RMPs, which can allow the implementation of general approaches and utilize its offline nature for using algorithms to generate paths in a faster and optimized way, therefore improving process quality while avoiding infeasibilities as demonstrated in this research. Other recent methods such as Programming by Demonstration (PbD), sensor-based, and augmented/virtual reality are still not robust enough for gaining acceptance in industrial scenarios.

Existing robot programming methods can be enhanced with the formal description of knowledge and the use of algorithms, which allow the automation of the path generation processes. A common denominator for the automatic generation of robot programs is the variety of infeasibilities that can appear through the execution of a process. Also, the optimization of movements for improving manufacturing process quality is important for the current production scenario. Due to these requirements, optimal sample-based algorithms are explored in this research for solving the proposed problem, while

also taking into account that these types of algorithms allow for the probabilistic exploration and optimization.

Several authors confirm these evaluations. Schraft and Meyer (2006, p.1) affirm that “SMEs currently do not often use robots in their production; due to the complex handling, especially the time-consuming programming”. Pan, Polden et al. (2012, p.87) assert that “although an automated flexible production cell is an intriguing prospect for small to medium-sized enterprises in current global market conditions, the complexity of programming remains one of the major hurdles preventing automation using industrial robots for SMEs”. Polden (2014, p.iii) states also that “programming robotic systems to carry out industrial manufacturing processes is often a difficult, time consuming, and costly exercise task”. Denkena, Wörn et al. (2004) also point out that the cost of programming the robot and its consequences, such as the teach-in, is a neglected aspect of robot deployment and emphasize the need for improving effectiveness in programming robotic actions.

A critical challenge to robotic path generation is the lack of a generalized framework for generating optimized paths for improving the process quality. A generalized approach would be appropriate taking into account the variety of process in which robots could be used. Knowledge models are not procedurally interpreted in order to allow the robotic path planning. Stand alone optimized solutions have been proposed in literature without them being a part of a general approach. Main optimizations have been focused on the transition paths of the process and not on the paths performing the manufacturing process itself. Also, the quality of the manufacturing process is directly related to the robotic paths, which emphasizes the importance of optimizing them. Bolmsjo (2014, p.4) also realizes this challenge when expressing that “automatic programming is a challenge for generic applications in industrial environment where it is important to achieve performance and quality with a high degree of certainty”.

Based on the above, optimized, efficient, automatic, and intuitive robotic programming is still a challenge in robotic manufacturing and one of the main reasons why robots are not widely implemented in SMEs for manufacturing purposes. In order to effectively and efficiently respond to current product variability requirements, face the lack of qualified personnel for programming robots, improve the robotic manufacturing process quality, reduce the required programming times, and simplify the programming complexity, SMEs must be provided easy and optimized path planning for robotic manufacturing systems as a contribution to achieve profitable and rapid changeover.

The conception and implementation of an automatic optimized path generation for robotic systems, which satisfies the process constraints and optimizes process criteria, can improve the robot programmability and process quality in manufacturing scenarios. This will lead to the implementation of automatic and simple-to-use tools for generating optimized robotic paths, which address the better configuration of manufacturing processes using industrial robots. Such contribution to the state of the art will enhance the changeover, which is of relevant importance to the competitiveness of industries and the SME sector. Based on the motivations and in the generalized needs of the robotic manufacturing community, a summarized problem statement and the research question is given below:

Summarized Problem Statement and Research Question

The generation of optimized paths for robotic manufacturing processes is currently time-consuming, complex, expert-dependent, and its optimization is primarily manual, and case-specific. Based on this deficiency, the following research question is proposed and will be solved in this thesis:

How to automatically generate offline optimized paths in reaction to infeasibilities for Robotic Manufacturing Processes (RMPs) under constraints and process optimization criteria?

1.3 Thesis Statement and Objectives

In order to solve the above-stated problem, the following thesis is proposed:

Robotic paths for manufacturing process can be optimized offline by implementing inputs based on:

- state-of-the-art models of products (geometric description), knowledge-based description of the process, and robot models (e.g. kinematic models),
- criteria for optimization of robotic manufacturing process,
- and specification of the robotic manufacturing task.

And by automatically configuring optimal sample-based path generation algorithms based on the procedural interpretation of the RMP; where the RMP is modeled into its Product, Process, and Resource (PPR) components and implemented into a generalized architecture.

To defend this proposition, the following objectives are proposed:

- *First objective*

Conceptualize and develop a model-based description of robotic manufacturing processes while taking into account the state of the art. This model-based description must offer a clear, structured, coherent, and computer-readable understanding of manufacturing processes performed with industrial robots and must be conceived and programmed in order to describe different processes. Moreover, this model-based description must allow the use of information and characteristics of the manufacturing process into robotic path generation algorithms.

- *Second objective*

Conceive and implement an approach and architecture in order to automatically configure path generation algorithms and characterize the RMP process using the PPR model for allowing its optimization. The automatic configuration must satisfy the constraints and allow the exploration of the process Degrees of Freedom (DoFs) in relation to the manufacturing task. This automatic configuration is intended as a novel way of representing and dealing with robotic manufacturing process tasks. It must allow the offline computation of optimized paths for achieving processing with industrial robots by allowing the definition of optimal criteria in the required spaces, such as the Cartesian and joint space.

- *Third objective*

Implement, experiment, and evaluate the proposed model-based description, the approach, and the architecture for generating optimized robotic paths in different manufacturing processes

while optimizing process related criteria. Robot programs with optimized movements will be the output of the developments performed under the second objective and will serve as input for the evaluation on robot cells. The approach, architecture, and implementation will be experimentally evaluated with the optimized collision avoidance in robotic welding, the optimized stiffness in robotic milling, and the optimization in simulation of the sensing strategy for a sensor-guided deburring process.

1.3.1 Focus of the Thesis and its Delimitation

The main focus of this dissertation is to approach the research question with the purpose of offering solutions to SMEs regarding the use of industrial robots in manufacturing processes for improving the required programming time and the final process quality. Consequently, the focus is to simplify the programming and at the same time to optimize paths by exploring the intrinsic functional redundancy of manufacturing processes, with the ultimate goal of improving the programmability and the manufacturing quality. Moreover, attention is given to the modularity of the approach in order to conceive a general architecture and improve the changeability of robot programming systems.

Due to the focus on SMEs, it is established that serial rotatory joint industrial articulated robots, as defined by the International Organization for Standardization (2012), IFR Statistical Department (2017), are the type of robots to be studied in this thesis. This category of robots is currently the most used type of robots in industry; and according to the IFR Statistical Department (2017, p.108), the share of total supply in 2016 was 61 %. Moreover, the focus is delimited to manufacturing processes for products with one dimensional features, such as lines, circumferences, or curves. Such processes represent an important share of the overall industry and therefore a high-level of priority and attention is given to them. Furthermore, in order to address straight-forward implementation by SMEs and to provide competitive advantage to commercial industrial robot controllers, processes with constant linear velocity will be the primary focus of attention. These processes also represent an important share of the current required processes in industry. Example of processes in which these features and characteristics are to be observed are the robotic welding, milling, or deburring.

The focus of this contribution is also centered on the path generation for the actual process execution, which has not been deeply studied in literature, in contrast to the sequence optimization or the task planning of transitions and auxiliary motions/paths in free-space which have been more widely researched. Due to the goal of offering solutions for SMEs, applied experimentation, and easy applicability must be conducted with robot cells and manufacturing processes. In order to verify the concept, the proposed approach, architecture, and implementation must allow computations in reasonable time for demonstrating the relevance and usefulness in industrial applications. However, it is worth mentioning that the focus of the thesis is not on improving the computational time of the implemented solution.

The approach and architecture must demonstrate that the automatic generation of optimized paths can be conducted for different processes by interchanging models. However, the focus of this thesis is not the further modeling of RMPs; instead, the focus is on using models that already exist in the literature setting them into the proposed PPR model.

1.4 Structure of this Thesis

This thesis is organized as follows:

- *Chapter 2: Fundamentals and State of the Art*

The basic foundations of this thesis are given in this chapter. Four main knowledge areas on robotic manufacturing are reviewed: (1) The robotic path generation, (2) the knowledge representation for RMPs, (3) the offline programming software, and (4) the path generation in the use-cases evaluated in this contribution. This chapter concludes with a proposition based on the analysis of the state of the art (refer to Section 2.7).

- *Chapter 3: The Automatic Optimized Offline Programming Approach for Robotic Manufacturing*

This chapter starts with the introduction of the proposed model-based description of the RMP in Section 3.1. Section 3.2 describes the robotic manufacturing process configuration space, which is a required concept for the definition of the approach. Section 3.3 formally defines the problem solved by the approach (refer to Section 3.4 with an extension described in Section 3.5) and the implemented architecture (refer to Section 3.6). The development of this chapter addresses the fulfillment of the first and second objective (refer to Section 1.3).

- *Chapters 4, 5, and 6: Implementation of the Use-Cases, Experimental Investigation, and Results*

These chapters explain the implementation of the optimized path generation approach in the investigated research use-cases. The evaluation and analysis of each implementation are described in detail for each use-case. The use-cases implemented and evaluated are (1) the optimized collision avoidance in robotic welding (refer to Chapter 4), (2) the optimized stiffness for robotic milling (refer to Chapter 5), and (3) the optimized sensing strategy for sensor-guided robotic deburring (refer to Chapter 6). Moreover, the automatic close-to-optimal workpiece positioning is also implemented for the welding and milling use-cases. These chapters approach the development of the third objective (refer to Section 1.3). For all use-cases, a similar structure as in Chapter 3 is addressed in order to illustrate the generalized proposed approach of this research.

- *Chapter 7: Conclusions and Outlook*

The conclusions and outlook of this work, outlining the potentials of the approach, are provided in this chapter.

2 Fundamentals and State of the Art

The following chapter presents the main methods and algorithms (refer to Section 2.1) used for the robotic path generation for manufacturing processes. A qualitative comparison of the methods and algorithms is performed in order to prioritize them with respect to the stated problem. The next section is dedicated to the knowledge representation in robotic manufacturing considered a key component of the automatic programming of robotic paths (refer to Section 2.2) followed by a review of some OLP software for industrial robots (refer to Section 2.3). The next three sections of this chapter (2.4, 2.5, and 2.6) give the state of the art of the implemented use-cases. To conclude, Section 2.7 presents a proposition based on the analysis of the overall state of the art and describes and defines the problematic in the robotic manufacturing processes studied.

2.1 Path Generation for Robotic Manufacturing Processes

According to its definition, a robot is an “actuated mechanism programmable in two or more axes with a degree of autonomy, moving within its environment, to perform intended tasks” (International Organization for Standardization 2012, p.2). Taking this into account, the path, trajectory, and motion generation problem are key issues in robotics and therefore these topics have been intensively studied and researched by the community.

The path is defined by the International Organization for Standardization (2012, p.14) as an “ordered set of poses” and the trajectory is a “path in time”. Current robot controllers available in the market provide well-developed and researched trajectory control and motion planning modules (both as defined by the International Organization for Standardization (2012)) to control the movement from point to point of a manipulator.

In the literature, the term planning is used for the calculation of paths as well as for the computation of sequences of paths or motions in order to achieve a certain task. In this dissertation, the term planning or generation is used as defined by the International Organization for Standardization (2012) for the computation of the actual path, which is the topic of study. If specific reference to the process sequencing is required the term "task planning" will be used, which is defined by the International Organization for Standardization (2012, p.30) as the “process of solving the task to be carried out by generating a task procedure which includes subtasks and motions”. An example of compilation of algorithms and developments related to the sequencing and scheduling in robot cells is provided by Dawande, Geismar et al. (2007). Note that focus of this work is the "path planning" and not the "task planning".

Important developments and technological achievements have been performed in the area of trajectory generation of point-to-point and multi-point trajectories (point-to-point trajectories with interpolation and approximation). These achievements are nowadays state-of-the-art technologies which are

commercially available for controlling actuation systems such as servo-drives in order to command a manipulator in space and time (Biagiotti and Melchiorri 2008). These technologies, accessible with each commercial robot controller, are relevant for the presented work because they are used to run the generated robot paths. However, note that the developments of this thesis are related to the "path planning" and not the "trajectory generation".

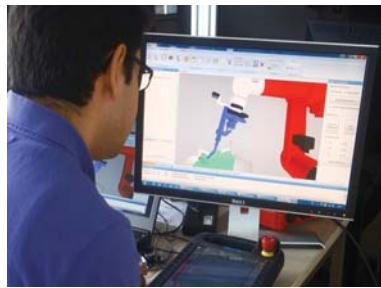
The path generation/planning problem has been intensively researched as well. Several methods and algorithms have been proposed and implemented for solving this complex task. This section presents the state of the art with respect to this topic by outlining the different variants of methods and algorithms available for solving the problem in robotic manufacturing applications.

2.1.1 Robot Programming Methods

The following subsection presents different programming methods available for robotic manufacturing processes. They are classified into: manual programming, offline programming, programming by demonstration, sensor-based programming, and programming using augmented/virtual reality. Methods combining the previous classifications are also possible programming variants. Figure 2.1 depicts these methods. The description of each method is offered below.



(a) Manual programming.



(b) Offline programming.



(c) Programming by demonstration.
Photo (Meyer 2011).



(d) Sensor-based programming.
Photo (Fraunhofer IPA, Rainer Bez).



(e) Programming using virtual reality. Photo (Schmirgel, Zimmermann et al. 2007).

Figure 2.1: Robot programming methods.

- *Robot Manual Programming*

Manual programming are the methods that require the end-user to create a program directly on the robot. As defined by Biggs and Macdonald (2003), this kind of programming can be

segmented into text-based and graphical. Text-based programming uses controller specific, generic procedural, or behavior-based languages for generating manually code. Graphical programming offers visualization environments for generating these programs with a direct relationship to the generated code instructions (Biggs and Macdonald 2003).

Today in industry, the most common practice of manual programming is the teach-in approach, lead-through, or teach programming (as officially defined by the International Organization for Standardization (2012)). Here the end-user holds a control device which allows the movement of the robot Tool Center Point (TCP), flange, or joints in different coordinate systems for saving robot poses textually or graphically for generating a robot program addressed to perform the task. Figure 2.1a depicts this programming method.

- *Robot Offline Programming (OLP)*

The OLP is defined as the “programming method where the task program is defined on devices separated from the robot for later entry into the robot” (International Organization for Standardization 2012, p.18). This method of programming uses a three dimensional (3D) simulation of the robot cell components. Geometrical features are used in order to teach poses to the robot. The simulation of the robot movements allows the end-user to identify drawbacks, such as collision, and to modify generated paths.

The main advantage of this method is that the operation of the real robotic system does not have to be interrupted during the initial programming phase. The main drawback of this method is that generated paths are not very accurate due to the differences between the geometries and locations of the elements in the simulated and the real scene (Kuss, Ksensow et al. 2017). A further deficiency is that OLP bases its programming just in geometric approaches, which is the reason why infeasibilities have to be treated manually. Geometrical approaches in OLP are methods, heuristics, or algorithms based just on the geometries of the simulated objects to generate paths. A detailed investigation of some of the current available OLP software for robotic manufacturing is provided in Section 2.3. Figure 2.1b illustrates this programming method.

- *Robot Programming by Demonstration (PbD)*

The robot PbD or lead-through is the method to teach a path or motion to a robot by demonstrating the task instead as using machine commands for creating the program. As with the offline programming method, one of the main goals of this approach is to avoid direct code programming. A detailed description and the history of robot PbD has been offered by Billard, Calinon et al. (2008). As a brief summary of this description, initial implementations used symbolic reasoning for teaching primitive actions into point-to-point movements. Then, state actions were encoded into graph-based representations for allowing symbolic reasoning to modify the nodes of the graph. Later on, several machine learning techniques were introduced for recognizing elementary operators and the generalization of a task. In these ways, several applications have been developed using different interfaces such as vision, data gloves, laser range finder, or kinesthetic teaching (Billard, Calinon et al. 2008).

In industrial environments, several applications of PbD have been reported. For example, the Fraunhofer Institute for Manufacturing Engineering and Automation (IPA) developed procedures

and methods which constitute the basis for handling and manufacturing processes demonstrating it in a welding application (Helms 2007). Meyer (2011), reports an increase of 45 % time improvement among four users while programming a robotic Metal Active Gas (MAG) welding task comparing a developed PbD approach and the manual programming with the robot panel. Embedding of learning approaches into PbD robotic systems have been also reported in industrial environments for improving programmability. An example is presented by Hollmann, Rost et al. (2010) which uses hidden Markov models for adapting user and process features. Further institutes have also implemented PbD methods for robotic programming. For instance, Pires, Veiga et al. (2009) presented a coworker scenario for SMEs and referred some related applications. A survey of robot learning from demonstration has been presented for robotic assembly operations (Zhu and Hu 2018). Figure 2.1c describes this programming method.

- *Sensor-based Robot Programming*

The following classification notates the applications in which additional sensors to the robot are used for supporting the programming of the manufacturing tasks. Common examples of these applications in industry are the joint edge detection and joint seam tracking which have been mainly implemented using contact and two dimensional (2D) laser scanners. For instance, Kah, Shrestha et al. (2015) present a survey of different sensor and systems available for the welding application. Applications using 3D sensors by matching the taken point cloud with the Computer-Aided Design (CAD) models of the product have been also lately presented in the literature (Kuss, Dietz et al. 2017).

The programming of the manufacturing task using this method is commonly supported by CAD formats of the product or by geometric specifications. The end-users initially locate the robot close to the feature intended to be manufactured for then configuring some parameters of the sensor-based approach. Final programs are generated by obtaining information from the sensors or by compensating errors during execution. This method emerged mainly in reaction to the inaccuracies presented in manual and OLP methods due to product tolerances, robot dependent errors, and the geometric mismatching between the modeled and the real robot cells. The sensor-based programming can generate the programs before the execution of the robotic manufacturing task or compensate deviations during the process itself in the Cartesian space. Figure 2.1d depicts this programming method.

- *Programming using Augmented/Virtual Reality*

Augmented and virtual reality techniques have evolved during the last years allowing developers and researchers to implement applications in several fields including the programming of robots and manufacturing tasks. Both techniques use computer generated information for facilitating the programming process. Augmented reality “overlays computer generated graphics, text, and three dimensional (3D) models over real video stream. Virtual information is embedded into the real world, thereby augmenting the real scene with additional information” (Ameri, Akan et al. 2010, p.56). Virtual reality “provides an indirect experience by creating a virtual space that interacts with the human sensory systems and overcomes spatial and physical constraints of the real world” (Choi, Jung et al. 2015, p.40).

The field of augmented/virtual programming in robotics is relatively new in comparison with approaches such as the manual and the OLP. Choi, Jung et al. (2015) review virtual reality applications in manufacturing industries in which some Industrial Robot (IR) solutions are observed. Some examples are offered by Ameri, Akan et al. (2010) who discuss some virtual reality applications in industrial environments using robots and present an application. Also, Schmirgel, Zimmermann et al. (2007) present an augmented and a virtual reality demonstrator using a Light Weight Robot (LWR). Figure 2.1e illustrates this programming method.

This subsection presented the main methods used for the robotic path generation for manufacturing processes. These methods are qualitative compared in the next subsection (2.1.2) with respect to how good these can solve the stated problem (refer to Section 1.2).

2.1.2 Comparison among Programming Methods

Different methods are available for programming robotic manufacturing tasks and have been categorized and reviewed in Subsection 2.1.1. According to the observation of applications in industry, it can be mentioned that the manual approach is the most used one in the SME sector. This assumption is also shared by other authors such as Pan, Polden et al. (2012, p.88) who state that the “online programming is still the only programming choice for most SMEs”. From the literature review, it is observed that even though the scientific community has achieved important steps in the conception of new methods; these are not widely used in industrial manufacturing environments and that just single use-cases are referred in literature. Alartartsev, Guedemann et al. (2012, p.365) also share this view when stating that “After analyzing different programming methods for industrial robots for the last 20 years, we have come to a conclusion that modern approaches in computer programming are not actually applied in industrial programming”.

As part of the methodology, a qualitative comparison between methods is offered in this subsection in order to illustrate which of the methods is more suitable for solving the stated problem: How to automatically generate offline optimized paths in reaction to infeasibilities for RMPs under constraints and process optimization criteria? The comparison is done based on the state of the art and on observations from industrial implementations through expert judgment by the author. Five different technical criteria are defined for the comparison. These criteria have been defined taking into account their relevance with respect to the stated problem. The definition of the criteria and evaluation for each method are explained below. A scale for the evaluation is proposed as follows: well-fulfilled, partially-fulfilled, and poorly-fulfilled. Table 2.1 summarizes the assessment.

- *Computation of Optimality*

The computation of optimality is defined as the ability- i.e. facility, efficiency, and effectiveness- to compute optimized paths. Considering the nature of the question addressed in this work, this is defined as a relevant criterion. The evaluation is based on the possibility for optimizing a given criterion.

For the manual programming method, the optimization depends purely on the programmer. As stated by Pan, Polden et al. (2012, p.87) “the quality of the program is limited by the skills of the operator and once the program is generated, it is very difficult to make further

amendments”. Similarly, in the PbD method the final quality of the program relies on the programmers demonstration expertise. Sensor-based approaches also rely the quality on the programmers’ specification of the relation between sensor, product feature, and robot reaction. For that reason, the qualification of poorly-fulfilled is given to these methods.

For methods in which an extrinsic programming approach is used (performed on devices separated from the robot), the option of computing optimality is of easier application. Examples of this are almost all the references presented in the Subsections 2.4.4 and 2.5.3 in which state-of-the-art optimizations for the robotic welding and milling are presented and which are based on OLP systems. The intrinsic characteristic of having a simulated environment allows the implementation of algorithms for testing different configurations and for applying optimal criteria. Due to this, the well-fulfilled qualification is then given to the OLP approach. Although, literature of optimization using augmented or virtual reality methods was not found during the state-of-the-art review, this method is also based on an extrinsic programming which could allow iterative offline computations. Nevertheless, the optimization using augmented or virtual reality would require higher computational performance due to its complexity in comparison to the OLP. Therefore, the partially-fulfilled qualification is granted to this method.

- *Automatic Programming*

The criterion of how well each method allows automation is also evaluated. A complete automatic system is defined as a system which does not require human intervention. Due to its nature, the manual, and the PbD methods are qualified as poorly-fulfilled because they always require the intervention of the operator.

Methods such as OLP, sensor-based, and augmented/virtual reality require some human configuration such as robot kinematics and objects in the scene, sensor-process parametrization, and task definitions. Furthermore, the CAD and sensor models should be provided for their operation. However, current research is pointing out the automation, simplification, and improvement of the intuitiveness of these methods (Leali, Pini et al. 2013). In comparison with the manual and the PbD method, these methods could be automated to a higher degree. Due to these reasons, a partially-fulfilled scale is given to these methods.

- *Accuracy*

The accuracy criterion is defined as the robot quality of being precise positioned with respect to the manufacturing process. This is a relevant issue when programming robots for manufacturing which influences the process quality.

The sensor-based approach reaches relevant accuracy due to the compensation based on the sensor feedback. Sensors used in this method are industrial oriented and therefore considered accurate. Several examples of industrial implementations for the welding process as example are reported by Kah, Shrestha et al. (2015). For this method, a well-fulfilled rank is adopted.

The manual approach offers relative accuracy depending on the programmer. An expert programmer can manipulate the robot to the desired position and orientation for compensating workpiece inaccuracies and robot-dependent errors. However, this compensation is limited to human accuracy. Due to this, the partially-fulfilled rank is given to the manual approach.

The PbD could also be considered accurate due to the use of sensor feedback. However up to date, no specification of the reachable accuracies is provided. These accuracies depend on the measurement systems and the learning effectiveness of the demonstrated paths. Moreover, the accuracy in implementations such as the walk-through approach or the use of cameras for recording human movements depends on the demonstration itself. Therefore, a partially-fulfilled score is also provided.

Methods which poorly-fulfill the accuracy criteria are the OLP and the augmented/virtual reality. Mismatching between the modeled and the real robot cells is the reason. In manufacturing environments, tolerances are narrow and also the exact positioning of the robot, product, and additional objects are crucial for achieving pose accuracy and process quality. Due to the operation in a virtual environment, this information does not match the reality causing that these methods are qualified with poorly-fulfilled.

- *Proven-in-use*

The proven-in-use criterion is related to the evidenced sufficient operation of the approach. This criterion is of importance because final implementation in real robots cells is intended as an objective of this study. Manual programming and OLP are widely used approaches in the industry and can be defined as simple and robust technologies (less involved complex systems and processes) in comparison to the other approaches. Consequently, the rank of well-fulfilled is given to these methods.

The sensor-based methods are also available in the industry and have been adopted if accuracy is demanded. Similarly, the augmented/virtual reality has been rapidly developed over the last years giving this technology a relevant level of maturity. For these two methods, the qualification of partially-fulfilled is provided mainly because these processes use additional hardware such as sensors and 3D glasses which compromise the robustness in some range. Moreover, these methods were introduced and developed after the manual programming and OLP supporting also the evaluation. Finally, the PbD is not well-established because this method is relative new and non-proven robust in manufacturing scenarios which justifies the qualification of poorly-fulfilled.

- *Safety during programming*

In current industrial environments, the safety is a crucial topic when implementing automatic systems. Thus, it is also an established criterion on the technical aspects to be evaluated during the programming in this subsection. Casualties using industrial robots must be prevented and in consequence several technologies such as safety fences and safety modules are available when programming robots in order to overcome risks.

The OLP and augmented/virtual reality methods relay on extrinsic programming, thereupon the interaction with the robot-machine is just required in case of needing over "teach-in" the path for compensating inaccuracies. The same "teach-in" approach is required for manual programming and it is nowadays strongly regulated. Just specific teaching modes, constraining the velocity, are allowed for programming the robot in this way. Therefore, a partially-fulfilled range is assigned to the previous methods. Other methods use intrinsic sensors and feedback to interpret the intentions of the end-user, recognize the objects in the scene, or sense the characteristics of the

process in order to generate programs. Therefore, they are more complex to keep safe. Regardless of the advances of these technologies, these methods are qualified with a poorly-fulfilled scale.

The proposed scale for the evaluation is represented as follows:

- meaning well-fulfilled,
- ◐ meaning partially-fulfilled, and
- meaning poorly-fulfilled.

For each scale, the numerical evaluation 5, 3, 1 is assigned respectively in order to assess the methods. Table 2.1 summarizes the comparison among these methods, presents the average of the technical criteria, and depicts the evaluation showing that the most suitable method to use and further develop in order to solve the proposed problem is the Offline Programming (OLP). Mainly, the facility to compute optimality in comparison to other methods makes the OLP preferable. It is worth mentioning that, the previous statement does not imply that this method is better than the others in general. Instead, the analysis focuses on how good the methods can solve the proposed problem (refer to Section 1.2).

Table 2.1: Comparison of programming methods in robotic manufacturing.

Scale: ● = well-fulfilled = 5; ◐ = partially-fulfilled = 3; ○ = poorly-fulfilled = 1

		Programming methods				
Criteria		Manual Programing	OLP	PbD	Sensor-based	Augmented Reality
Technical	Optimality Computation	○	●	○	○	◐
	Automatic Programming	○	◐	○	◐	◐
	Accuracy	◐	○	◐	●	○
	Proven-in-use	●	●	○	◐	◐
	Safety	◐	◐	○	○	◐
Average All Criteria		2.6	3.4	1.4	2.6	2.6

2.1.3 Robot Path Generation Algorithms

The following subsection presents the different algorithms available for generating paths in robotic manufacturing processes. They are generally classified in this contribution as: grid-based, computational mathematical, bio-inspired, and sample-based algorithms. A similar general classification was described by Yang, Qi et al. (2016). Approaches combining the previous classifications are also possible programming variants. A description of each algorithm is offered below:

2.1.3.1 Grid-based Search

These kinds of algorithms are based on the use of a grid for computing a path. Some authors refer these algorithms such as roadmap methods (Latombe 1991). The grid is a set of discrete nodes through which the algorithm leads the robot to a goal node moving itself to an adjacent node. The most recognized algorithms are the Dijkstra algorithm, the A-star algorithm (A*), the visibility graph, the planning over a cell decomposition, and the planning over a Voronoi diagram (Latombe 1991).

The Dijkstra algorithm finds the optimal path in a grid by evaluating in each node the best connection to all adjacent nodes successively up to the point of reaching the goal node. The A* is an evolution of the previously described algorithm. The novelty in comparison with its predecessor is that it includes a heuristic estimate of the cost to get to the goal node. Therefore, it considers first the options that lead in the fastest way to the solution. This gives the reason, why it is also considered an informed search algorithm. The visibility graph “mainly applies to two-dimensional configuration spaces with a polygonal C-obstacle region. The visibility graph is the non-directed graph whose nodes are the initial and goal configurations and all the C-obstacle vertices” (Latombe 1991, p.12). The cell decomposition “consists of decomposing the robot’s free space into simple regions, called cells, such that a path between any two configurations in a cell can be easily generated” (Latombe 1991, p.14).

In the literature for industrial robotics and robotic manufacturing processes, several applications are reported using these algorithms. Verbarg (1998) presents an approach using the A* algorithm with clearance computation and a denominated sparsity heuristic for computing not necessarily optimal collision-free paths conceived for robotic welding. Halperin, Latombe et al. (2000) present a framework for finding motions in order to assemble products by defining a motion space by constructing different graphs related to the constraints of the assembly process. Hein (2003) formulates an automated collision-free algorithm for industrial robots based on the A* search in the space spanned by the robot joints with hierarchical distance computation based on the given CAD model improving results optimizing the computed paths by the A* algorithm.

Besides, Thomas, Barrenscheen et al. (2003) present an assembly sequence planning by projecting the C-space of obstacles using the stereographic projection for building a graph used for the planning phase. Denkena, Wörn et al. (2004) describe the advantage of using planning algorithms in industrial scenarios, such as the hierarchical A*, which divides the grid in a hierarchical way. Further references such as (Verbarg 1997, 1998, Ting, Lei et al. 2002, Erdös, Kemény et al. 2013, Polden 2014, Larsen, Pham et al. 2015, Larkin, Short et al. 2016, Spensieri, Carlson et al. 2016a) relate the use of this categorization of algorithms for the robotic welding process and collision avoidance. Review of these publications are given in Section 2.4.

2.1.3.2 Mathematical Computation Algorithms

Mathematical computation algorithms as defined by Yang, Qi et al. (2016, p.4) “include linear algorithms and optimal control. These methods model the environment (kinematic constraints) as well as the system and then bound the cost function with all the kinematic and dynamic constraints bounds which are inequalities or equations to achieve an optimal solution”. This sort of algorithms approaches to model the robot, task, and environment in a mathematical way; for instance using differential equations.

On the robotic manufacturing side, some implementations are to be observed in the literature. On Subsection 2.4.1, some works for the robotic welding using these algorithms are presented (Huo and Baron 2005, Bayan, Pereira et al. 2005, Gerdts, Henrion et al. 2012, Björkenstam, Spensieri et al. 2014, Landry, Gerdts et al. 2014). Also, Subsection 2.5.3 refers some publications using mathematical computation algorithms for the robotic milling as example (Makhanov, Batanov et al. 2002, Owen, Croft et al. 2008, Xiao and Huan 2012, Subrin, Sabourin et al. 2013, Mousavi, Gagnol et al. 2017).

2.1.3.3 Bio-inspired Algorithms

As defined by Yang, Qi et al. (2016, p.5) bio-inspired algorithms “originate from mimicking biological behavior to solve problems. This kind of planning methods leaves out the process of constructing complex environment models to search a near optimal path based on stochastic approaches”. Two main subcategories are also defined by the same author: the evolutionary algorithms and the neural networks.

Some of the most used bio-inspired algorithms in planning are the genetic algorithms, neural networks, reinforcement learning, and the ant colony optimization. Due to the latest developments of these algorithms and the power of computation of computers more implementations on industrial scenarios are to be observed in the literature. Examples of applications for the robotic welding are provided in Section 2.4 with the following publications (Mitsi, Bouzakis et al. 2008, Liao, Wang et al. 2010, Wang, Shi et al. 2015, 2016, Doan, Tao et al. 2016).

2.1.3.4 Sample-Based Algorithms

A detailed description of these algorithms and their fundamentals is offered because these algorithms are identified as the best suited (refer to Subsection 2.1.5) to answer the proposed research question (refer to Section 1.2).

Sample-based algorithms started to be developed when complex path problems were approached. Basic foundations remote to the introduction of the Configuration Space (C-space) by Lozano-Perez (1983). The C-space is defined as follows,

Definition of Configuration Space (C-space)

As defined by Lynch and Park (2017, p.12) “The C-space is the n-dimensional space containing all possible configurations of the robot. The configuration of a robot is represented by a point in its C-space”.

As defined by Latombe (1991, p.xi) “the geometry of the task can be mapped in this configuration space. Physical concepts, such as force and friction, can also be represented in this space as additional geometrical constructs”. In contrast with the grid-based search, these sort of algorithms do not require a previous mapping of the grid. These algorithms explore the C-space by sampling states which are the specification of one transformation applied to the rigid body or robot in this C-space. The state (denoted as s) is also referred in all the related literature as sample or configuration. The same denotations will be used in this thesis.

Several algorithms have been proposed under this category. These algorithms are classified in the literature as single-query and multiple-query. Single-query algorithms compute solutions, in case of existing, for just a pair of start and goal states. In contrast, multiple-query algorithms are able to find solutions for multiple pairs of start and goal states at the cost of representative computation for building the data structure. Some of the most representatives single-query algorithms are: The Rapidly-exploring Random Tree (RRT), Transition-based RRT (T-RRT), the RRT Connect (Kuffner and Lavalley 2000), the Lower Bound Tree RRT (LBTRRT), the Sparse Stable RRT, the Parallel RRT (pRRT), the Lazy RRT (LazyRRT), the Expansive Space Trees (EST), the Kinematic Planning by Interior-Exterior Cell Exploration (KPIECE), the Search Tree with Resolution Independent Density Estimation (STRIDE), the Path-Directed Subdivision Trees (PDST), the Fast Marching Tree algorithm (FMT*) and the Bidirectional Fast Marching Tree algorithm (BFMT*), among others. Some of the most representatives multiple-query algorithms are: Probabilistic Roadmap (PRM), its variant the Lazy PRM, the Spars, and its second version. A summary of several derivations of the PRM and the RRT is provided by Short, Pan et al. (2016).

Other algorithms which are categorized under sample-based are the potential fields. The potential fields do not require grids. As defined by (Latombe 1991, p.295) “the potential function is typically (but not necessarily) defined over free-space as the sum of an attractive potential pulling the robot toward the goal configuration and a repulsive potential pushing the robot away from the obstacles”. As examples, implementations of this algorithm for avoiding collision are referenced in Subsection 2.4.3 (Juang 1998, Csiszar, Drust et al. 2012).

As stated by several authors, the sample-based algorithms have emerged as an efficient and extended solution to the complex motion problems of robots (Elbanhawi and Simic 2014, Lavalley 2011, Choset, Lynch et al. 2005) in several applications (Tsianos, Şucan et al. 2007). Moreover, nowadays open-source software is available for implementing and evaluating further applications and the optimality of this technique has been further researched and in some of the cases proved. Having introduced the sample-based algorithms, the definition of path generation problem will be given. Authors such as Latombe (1991), Lavalley (2011) have similarly defined the path and the path generation problem as follows:

Formal Definition of Path:

The path is defined as the continuous function $\sigma : [0, 100] \rightarrow \mathbb{R}^n$, where n is the dimension of the path generation problem. The 100 constant represents the percentage of the path. The path is defined as feasible if $\sigma(\tau) \in \mathcal{S}_{free}$, for all τ in $[0, 100]$. Here the feasible region is denoted as $\mathcal{S}_{free}$ and the infeasible region as $\mathcal{S}_{unfree}$. Figure 2.2 depicts the path.

Formal Definition of the Path Generation Problem:

In a C-space or space (e.g. the 2D or 3D Euclidean space, respectively $\mathbb{R}^2$ and $\mathbb{R}^3$, or the Cartesian space $\mathbb{SE}^3$, as defined in (Lavalley 2006, p. 148)), with an infeasible region $\mathcal{S}_{unfree}$, given an initial placement of the robot or start state (s_s), compute how to gradually move the robot into a desired goal state (s_g) so that it never touches the infeasible region. Figure 2.2 depicts the path generation problem.

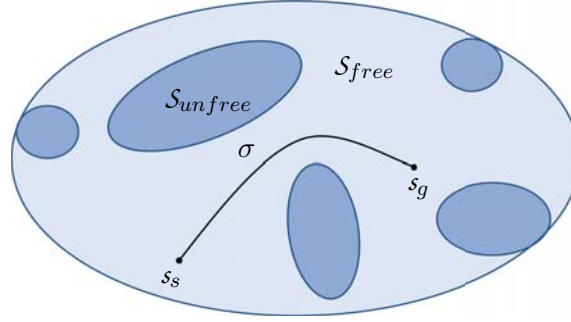


Figure 2.2: Configuration space with feasible and infeasible regions, start and goal states, and a path. Similar depicted as by (Lavalle 2006, p.157).

With respect to the planning using sample-based algorithms, an important topic is the constraining of movements. Some works are to be found in literature: Yao and Gupta (2005) present an approach for alternating between the task and configuration space in order to satisfy end-effector positions. Stilman (2007, 2010) constrains the planning in joint space representing constraints in the task space for later using sample-based algorithms in order to plan. Berenson, Srinivasa et al. (2011) outline similarly a framework for representing constraints which are later taken into account for the planning using sample-based algorithms. The previously presented approaches are focused on constraining non-manufacturing tasks and have been tested mainly for object manipulation or grasping which are applications in which sample-based applications are more widely used as noticed in the literature review.

On industrial implementations, some stand-alone applications using sample-based algorithms are reported in literature. For instance, on the robotic welding scenario, the works from Larkin, Short et al. (2016) and Polden (2014) use PRM and Lazy PRM respectively for planning the transition motions in free-space. Bohlin and Kavraki (2000) presented a performance test for a 6 DoF robot using RobotMaster for picking plane sheets of metal which should be positioned in a hydraulic press for being bent demonstrating that the Lazy PRM algorithm just required exploration of 0.1 % of the roadmap in comparison with its predecessor the PRM algorithm. Other applications are for instance the one presented by Maeda, Ushioda et al. (2008) who improved industrial robot programming of a pick and place task by manually demonstrating the volume sweeping and the planning using a single-query bi-directional probabilistic roadmap planner. After having introduced the sample-based algorithms, their optimal versions are described below.

Optimal Sample-based Algorithms

Optimization has been researched in many sciences and this is also the case in robotics. Important advances have been reached in the area of robot control which aims to map movements from the task space to the joint space in order to generate control functions, such as the signals required by the servomotors of an industrial robot for optimizing specific criteria. On the other hand “for global aspects of motion, it seems that optimality criteria is more problematic” (Toussaint, Ritter et al. 2015, p.381) and therefore optimized motion methods have been approached for overcoming the complexity of the problem.

A clear distinction between optimal and optimized motion has been explained by Laumond, Mansard et al. (2014). Optimal planning refers to finding the unique optimal solution, while optimized planning

refers to iteratively finding better solutions approaching to the unique optimum. As mentioned by Laumond, Mansard et al. (2014, p.82) “most of the time, robot algorithms aiming at computing an optimal motion provide in fact an optimized motion which is not optimal at all, but is the output of a given optimization method”.

The optimized approach has also been applied with the sample-based approach. After the appearance of the first algorithms and their successful implementation in some applications, the formal analysis of the solution quality returned by such algorithms was properly analyzed by Karaman and Frazzoli (2011) who also evolved the most relevant algorithms existing by the time, the PRM and RRT, and defined the path planning problem for finding a path with minimum cost as follows:

Formal Definition of the Optimized Path Generation Problem:

Given a path generation problem (as defined above) and a cost function $\mathbf{C}(\cdot) \geq 0$, find a feasible path σ^* , such that $\mathbf{C}(\sigma^*) = \min\{\mathbf{C}(\sigma) : \sigma \in \mathcal{S}_{free}\}$. If such path does not exist, report failure (Karaman and Frazzoli 2011).

After having introduced formally the optimized path generation problem, some relevant concepts which are later used in the description of optimal algorithms are introduced. Sample-based motion algorithms are not complete due to their probabilistic nature. “Even though these algorithms are not complete, they provide probabilistic completeness guarantees in the sense that the probability that the planner fails to return a solution, if one exists, decays to zero as the number of samples approaches infinity” (Karaman and Frazzoli 2011, p.846). “Moreover, the rate of decay of the probability of failure is exponential, under the assumption that the environment has good visibility properties” (Karaman and Frazzoli 2011, p.847). The characteristic of probabilistic completeness is of importance in the implementation and in practical scenarios because software should define if there are feasible paths.

Another important concept is the one of asymptotic optimality. This concept states that the longer a path generation algorithm runs, which means as longer the algorithm explores the C-space and its different configurations, the better the retrieved path with respect to a given cost $\mathbf{C}(\sigma)$ is. The notion of asymptotic optimality provided by some algorithms assures the convergence to an optimum which is one of the main characteristics explored in order to compute optimality. Marble and Kostas (2017, p.5) define asymptotic optimality in path planning as follows: “an algorithm is asymptotically optimal if, for any path planning problem and cost function $\mathbf{C}(\sigma)$ that admit a robust optimal solution with finite cost $\mathbf{c}^*$, the probability that it will find a solution of cost $\mathbf{c}^*(\sigma)$ converges to 1 as the number of iterations approach infinity”.

Another concept attained in this field is the one of asymptotic near-optimality and it has been defined from the area of graph planning and graph spanners by relaxing the optimum definition. As defined by Marble and Kostas (2017, p.7), “an algorithm is asymptotically near-optimal if, for any path planning problem and cost function that admit a robust optimal solution with finite cost $\mathbf{c}^*$, the probability that it will find a solution of cost $\mathbf{c}^* \leq \mathbf{t}\mathbf{c}$ for some stretch factor $\mathbf{t} \geq 1$ converges to 1 as the number of iterations approach infinity”. This relaxation has been introduced mainly in order to reduce computational complexity. The previously introduced concepts are important for the definition of some optimal sample-based planning algorithms. Some optimal sample-based algorithms are introduced below:

- *The Rapidly-exploring Random Tree* (RRT*)*

The RRT* algorithm is based on its predecessor RRT. The RRT algorithm was introduced by Lavalle (1998) as a randomized data structure designed for a broad class of path planning problems. In contrast with the PRM algorithm, the RRT is initialized with a graph that includes the initial state for later expanding a tree of feasible trajectories. The optimized variant of this algorithm or RRT* has been introduced by Karaman and Frazzoli (2011). This algorithm avoids the formation of cycles by removing edges that are not part of the computed better cost path over the tree.

- *The informed RRT**

The informed RRT* is an evolution of the RRT*. This algorithm was introduced by Gammell, Srinivasa et al. (2014) for improving the convergence rate and final solution quality by focusing the search estimating a subset of states that can improve a solution for problems seeking to optimize an specific cost. “This subset can be estimated by a prolate hyperspheroid (a special type of hyperellipsoid) with the initial and goal states as focal points. The probability of adding a new state from this subset through rejection sampling of a larger set becomes arbitrarily small as the dimension of the problem increases, the size of the sampled set increases, or the solution approaches the theoretical minimum.” (Gammell, Srinivasa et al. 2014, pp.3003)

- *The Batch Informed Trees (BIT*)*

The BIT* is an algorithm based on the unification of graph- and sample-based planning techniques introduced by Gammell, Srinivasa et al. (2015). The algorithm uses the ordered nature of graph methods and the anytime scalability of sampling-based methods. This algorithm has been shown to be probabilistic complete and asymptotic optimal (Gammell, Srinivasa et al. 2015).

- *The Transition-based Rapid-Random Tree (T-RRT)*

The T-RRT algorithm combines the exploratory strength of the RRT with stochastic optimization methods that use transition tests which accept or reject new potential states (Jaillet, Cortés et al. 2008). This planner allows the path planning considering the cost function defined over the configuration space following valleys and saddle points of the C-space costmap. The planner computes low-cost solutions with respect to a quality criterion based on the notion of mechanical work (Jaillet, Cortés et al. 2016). Bi-directional enhancement of this algorithm has also been presented by Devaurs, Siméon et al. (2013).

- *The Probabilistic Roadmap* (PRM*)*

The PRM* algorithm is based on its predecessor the PRM and the simplified PRM. The PRM planner consists of a planning method which is accomplished in two phases. The first one is the learning phase, in which a probabilistic roadmap is constructed as a graph with feasible states and edges computed with a local planner. The second one is denominated the query phase, which connects the configured start and goal state with two nodes initially for finding later a feasible path through concatenation (Kavraki, Svestka et al. 1996). The simplified modification initializes the vertex set and performs the same work as the PRM method, with the difference that it allows

connections between vertices in the same connected component with a predefined connecting distance.

The "star strategy" of the Probabilistic Roadmap (PRM) has been introduced by Karaman and Frazzoli (2011) proving the asymptotic optimality of the algorithm and its probabilistic completeness. This asymptotic optimality characteristic is gained due to the introduction of a function which decreases the connecting distance with respect to the number of samples and the dimension of the space.

- *The Lazy PRM**

The Lazy PRM* algorithm is based on its predecessor PRM. The Lazy PRM in comparison to the PRM minimizes the number of checks performed by the validity check function. This planner assumes initially all nodes feasible and searches for the shortest path. In case of finding infeasible nodes, these are removed from the path and the planner either searches for a new shortest path or updates the roadmap with new nodes for continuing finding a new shortest path (Bohlin and Kavraki 2000). Advantages of this planner is the less requirement of processing time because it does not explore the overall roadmap.

- *The Spars Roadmap Spanner*

The Spars Roadmap Spanner algorithm has been proposed by Dobson, Krontiris et al. (2013). This planning algorithm provides probabilistic completeness, near-optimality, and the probability of adding nodes to the spanner which converges to zero as iterations increase. In contrast with other planners, the near-optimality characteristic allows this algorithm to compute good quality paths with finite-size roadmaps showing that the relaxation of the optimality allows faster computation of close-to-optimal solutions (Dobson, Krontiris et al. 2013).

- *The Spars Roadmap Spanners Version 2.0*

The Spars V2 algorithm offers probabilistic completeness and asymptotic near-optimality as the previously introduced Spars method. As described by its author, "the improved method provides these properties by building two graphs in parallel: a dense asymptotically optimal roadmap based on PRM* and its spanner" (Dobson and Bekris 2013, p.4106). Experimental evaluation shows that this algorithm significantly reduced memory requirements upon construction versus other methods.

- *The Fast Marching Tree algorithm (FMT)**

The FMT* as stated by its author is an "algorithm which performs a lazy dynamic programming recursion on a predetermined number of probabilistically drawn samples to grow a tree of paths, which moves steadily outward in cost-to-arrive space" (Janson, Schmerling et al. 2015, p.883). This algorithm has been proven to be asymptotically optimal and it has been demonstrated that it converges to an optimal solution (Janson, Schmerling et al. 2015). Bi-directional search has been also implemented using the FMT* and the proof of asymptotically optimality with this planner modality has been demonstrated by (Starek, Gómez et al. 2015).

This subsection (2.1.3) presented the main algorithms used for the robotic path generation for manufacturing processes. These algorithms are qualitative compared in Subsection (2.1.5) with respect to how good these can solve the stated problem (refer to Section 1.2). The following subsection present some path planning libraries.

2.1.4 Path Generation Libraries

Advances in the area of robotic path generation facilitate today the access to different software packages. These software packages permit further development and testing of new robot applications and algorithms. Some of the existing libraries are: The "motion strategy library" (Lavalle 2003), the "robotics library" (Rickert and Gaschler 2017), the "Open Motion Planning Library (OMPL)" (Şucan, Moll et al. 2012) (also available in the application "MoveIt" in the Robot Operative System (ROS) (Coleman, Sucan et al. 2014)), "CuikSuite" (Porta, Ros et al. 2014), the "Open Robotics Automation Virtual Environment (OpenRAVE)" (Diankov 2010), the "Motion Planning Kit" (Stanford Univ. 2006), the "motion planning kernel" (Gipson, Gupta et al. 2001), the "search-based planning library" (Ferguson 2013), and the "Matlab robotic toolbox" (Corke 2011) which also provide some sample-based algorithms. A trajectory description conception has been intended for unifying the semantics required for interfacing some motion planning libraries (Alatartsev, Guedemann et al. 2012).

From these libraries, the OMPL is selected in this contribution because it is (1) among the current and continued maintained libraries, (2) it provides several planning algorithms of different kinds (multi-query, single-query, control-based, and optimal sample-based planners) and (3) it is a widely used library in the community. A detailed description of this library is offered by the Kavraki Lab (Kavraki 2017). In general, a problem definition class is provided by OMPL for specifying the start and goal states and for having a pointer to the generated path. Another class allows the specification of the planning algorithm. Moreover, the space information class gathers the specification of the C-space (StateSpace class), a validity check function, the motion validator, and a valid state sampler.

2.1.5 Comparison among Path Generation Algorithms

Several algorithms have been proposed for planning robotic manufacturing paths. Subsection 2.1.3 categorized and described them. With the aim to determine which kind of algorithms further explore for solving the research question, a qualitative analysis for comparing and prioritizing these categories is provided through expert judgment by the author.

The comparison evaluates the following technical criteria: Computation of optimality, modeling complexity, computational requirements, and configuration requirements. Each of this criteria is defined and evaluated below. These criteria have been defined taking into account their relevance with respect to the stated problem. The same scale as in the previous comparison is proposed: well-fulfilled, partially-fulfilled, and poorly-fulfilled. Table 2.2 summarizes the assessment.

- *Computation of optimality*

This criterion evaluates if the algorithms can compute optimal paths. If proper mathematical models are provided and proper optimization methods are used, it is possible to calculate

and compute the optimum. Therefore, the only algorithm evaluated with well-fulfilled is the mathematical computation.

Sample-based algorithms are on their basic versions, such a PRM and RRT, not capable of computing optima. However, on their latest evolution, these algorithms offer the characteristic of asymptotic optimality and near-optimality. Due to this characteristic, which assures an approximation to the optimum, these algorithms are evaluated with partially-fulfilled. Contrary, as stated by Yang, Qi et al. (2016, p.13) in the grid-based search “the nodes provide an incomplete structure of the configuration space. Therefore, this kind of method can only achieve the best result limited by the representation of environment”. This deficiency is shared in the literature and therefore a grade of poorly-fulfilled is assigned to this kind of algorithm.

Referring to bio-inspired algorithms, for instance in the case of genetic search, Larsen, Kim et al. (2017, p.246) mentioned that “really crucial for the success of the genetic algorithm is the fitness function” which should be defined meaningfully for the comparison of the different chromosomes. Meaning that the optimality depends on definitions of the problem itself which gives the reason for the evaluation of this algorithm with poorly-fulfilled.

- *Modeling Complexity*

Due to the nature of the mathematical computation algorithms, exact models are required for its functioning (e.g. differential equations). This kind of modeling is more complex, in some cases, than the modeling in a discrete fashion representing difficulties for the implementation. This can be illustrated with the collision avoidance in robotics (refer to Subsection 2.4.3). Due to the complexity of collision objects in real scenarios, it is complex to establish mathematical models for such structures. Implemented applications use instead discrete modeling for detecting the collision between objects in OLP. Due to this reason, the modeling complexity criteria for the mathematical computation is evaluated as poorly-fulfilled.

Instead, the sample-based and bio-inspired algorithms can function with discrete and/or continuous modules. As example, a sample-based algorithm can use a collision detection function based on discrete models of objects such as the Stereolithography (STL) formats and in synergy relay in a mathematical model of the robot kinematics. Thus, the modeling complexity is evaluated for these two algorithms as well-fulfilled. Grid-based algorithms are evaluated as partially-fulfilled because they require a grid and therefore a more detailed and complex model of the C-space.

- *Computational Requirements*

The criterion of computational requirements is evaluated due to its importance on the implementation phase and on the final effectiveness and efficiency of a planning system. Computational requirements for continuous mathematics, and mainly if it is related with optimization, are considered high. Nowadays, some state-of-the-art programming languages offer options to specify equations symbolically. These definitions and computations demand more resources. In relation to the preceding, the score of poorly-fulfilled is assigned to the mathematical computation.

Similarly, a deficiency of grid-based methods is that the complexity of the grid grows exponentially if the C-space increases its dimensions, incurring in the decrease of the computational performance

in high-dimensional problems. Lavalle, Branicky et al. (2004) point out the benefits of random sampling in comparison with grid-based methods when dealing with high dimensions. Therefore the poorly-fulfilled evaluation is assigned to the grid-based methods.

In regard to sample-based planning as argued by Tang, Khaksar et al. (2012, p.20) and reflected in the literature; “despite their simplicity, which is exemplified in the basic PRM planner, the sampling-based planners are capable of dealing with robots with many degrees of freedom and with many different constraints. Among other, the sampling-based planners can take into account kinematic and dynamic constraints, closed-loop kinematics, stability constraints, reconfigurable robots, energy constraints, contact constraints, visibility constraints, and others” which justify the degree of well-fulfilled.

Bio-inspired algorithms are also considered efficient with respect to computational requirements. However, a comparison between sample-based algorithms (RRT and T-RRT) and, for example, the genetic search method has depicted better performance for the sample-based algorithms (Larsen, Kim et al. 2017). Therefore, a qualification of partially-fulfilled is given to this category.

- *Configuration Requirements*

Previously required configurations can make the implementation and usage of algorithms difficult. Mathematical algorithms do not require any kind of configuration and therefore the best qualification (well-fulfilled) for this category is assigned.

Sample-based and grid-based algorithms require some parametrization for its proper functioning such as the discretization step of the C-space. Moreover, each algorithm requires specific configurations. The degree of partially-fulfilled is given to these categories.

Bio-inspired algorithms require also several parametrizations. For instance, as reported by Larsen, Kim et al. (2017), a genetic search for the planning of industrial robots requires the configuration of parameters such as individual count, number of iterations, single mutation rate, one-point crossover rate, random portion selection, population count, collision interpolation distance, and fitness weights which are more configurations than for other methods. Moreover, as also stated by Larsen, Kim et al. (2017), for this kind of algorithms, the definition of fitness function is of importance for finding a near-optimum which is also considered a configuration requirement. Due to this reason, the poorly-fulfilled degree is taken into account for the bio-inspired algorithms.

The proposed scale for the evaluation is also represented here as follows:

- meaning well-fulfilled,
- ◐ meaning partially-fulfilled, and
- meaning poorly-fulfilled.

For each scale, a numerical evaluation 5, 3, 1 is assigned respectively in order to assess and select the most suitable algorithm. Table 2.2 summarizes the comparison among algorithms, presents the average of the criteria, and depicts the evaluation, concluding that the most suitable kind of algorithms to use in order to solve the proposed problem are the sample-based algorithms. Mathematical algorithms allow the computation of the unique optimum with the disadvantages of being more particular

and requiring higher computational performance. Sample-based algorithms instead do not compute the unique optimum but allow the approximation to the optimum without compromising the complexity and the computational requirements.

Due to the previous given comparisons and taking into account that a general approach and architecture is desired, the sample-based algorithms are then further studied in this research. It is worth mentioning that, the previous statement does not imply that this category of algorithms is better than the others. Instead, the analysis justifies which sort of algorithms is more suitable for solving the proposed problem (refer to Section 1.2).

Table 2.2: Comparison of programming algorithms in robotic manufacturing.
Scale: ● = well-fulfilled = 5; ◐ = partially-fulfilled = 3; ○ = poorly-fulfilled = 1

Programming Algorithms				
Criteria	Grid-based	Sample-based	Mathematical computation	Bio-inspired
Optimality Computation	○	◐	●	○
Modeling Complexity	◐	●	○	●
Computational Requirements	○	●	○	◐
Configuration Requirements	◐	◐	●	○
Average Criteria	2.0	4.0	3.0	2.5

This section (2.1) presented the state of the art of the path generation for manufacturing processes describing the main methods and algorithms. A qualitative comparison between them was performed. Based on the comparison, the Offline Programming (OLP) method and the sample-based algorithms are defined as the most suitable to use and further develop in order to solve the proposed problem.

2.2 Knowledge Representation in Robotic Manufacturing

Knowledge representation and its semantics have been a field of constant research and development in robotics for manufacturing due to the complexity of the task that robots are intended to perform. The following section presents a structured overview on this field. A dedicated subsection (refer to 2.2.1) is offered to the PPR model due to its representative use in the robotic manufacturing context and its use in this research.

- *Spatial Descriptions and Transformations*

Since the early development of robotics theory, spatial descriptions and transformations in order to model robot kinematics and dynamics have been established, i.e. Craig (1989). Nowadays, these descriptions and transformations are widely adopted in robotics. Moreover, open-source

software can be used for describing these geometric and spatial relations between bodies, i.e. (Laet, Bellens et al. 2013a,b).

- *Formalisms*

As an extension to these descriptions, formalisms or formal descriptions in mathematical, logical, or semantical terms have also appeared in order to facilitate modeling of robot tasks and controls. For instance, the Task Frame Formalism proposed by Mason (1981) provided initial methodologies for synthesizing robot force control. In this work, a task was semantically modeled by describing the manipulator translation and angular velocity, the force and torque at the TCP, and the task geometry constraints. Further development of this formalism is extended to the manipulation of objects through compliant motion (Lozano-Perez, Mason et al. 1984, Kröger, Finkemeyer et al. 2004).

Later approaches have been proposed in order to reduce complexity when programming robots. De Schutter, De Laet et al. (2007) introduced a systematic constraint-based approach to specify complex tasks of general sensor-based robotic systems consisting of rigid links and joints. This approach integrates both instantaneous task specification and estimation of geometric uncertainty. This knowledge representation of tasks is the base for the derivation of the control structure proposed with iTaSC (Instantaneous Task Specification using Constraints). This control structure is proposed for satisfying constraints between the object's and feature's frames. In this way, relative robot motion between objects is dynamically controlled for accomplishing robot tasks (Rutgeerts 2007). An application of this approach has been reported by Laet, Decre et al. (2007). Later research has focused on the interpretation of geometric constraints for specifying robot control (Scioni, Borghesan et al. 2015, Borghesan, Scioni et al. 2016).

- *High-level Planning and Task Planning*

Parallel to the development of formalisms, extensive research has been performed also in the area of high-level and task planning. High-level planning uses symbolic states descriptors and operators that characterize the effects of actions over the descriptors. Two main formalisms are to be found in the literature: The set-theoretic representation and the classical representation (Ghallab, Nau et al. 2004). Moreover, the Planning and Domain Definition language has been proposed as the semantic modeling tool and it is extensively used for this sort of planning (McDermott, Ghallab et al. 1998).

On the industrial robotics site, several applications are to be found in the literature using high-level planning. Stenmark and Malec (2015), Stenmark (2015) outline a system for high-level programming with code generation and natural language understanding. Arnold (2008) presents an environment for offline programming of industrial robots with a high-level language, a declarative language, and a user front-end interface on case studies such as pick and place, palletization, and traffic control with a robot. Wolfe, Marthi et al. (2010), Choi and Amir (2009) describe approaches combining high-level planning, and motion planning for robotic manipulation applications. Stenmark and Nagues (2013) report a system, evolved from the core ontology Rosseta, for appropriating knowledge representation and reasoning services for industrial systems.

2.2.1 The Product, Process, and Resource Model in Robotic Manufacturing

The Product, Process, and Resource (PPR) model is a classification structure for managing modularity in production (Cutting-Decelle, Young et al. 2007). Therefore, the PPR model is a widely used concept on the knowledge representation for industrial robot applications. Each classification in respect to RMPs is reviewed in next subsections. This model is used for instance in the process planning phase (Nielsen 2003), on the exchange of plant engineering information (Schleipen and Drath 2009), in product life cycle management systems (Sang, Tae et al. 2010), on the modeling of industrial robotic systems cost (Dietz, Pott et al. 2015), the digital modeling and simulation (Agyapong-Kodua, Haraszko et al. 2014), the planning of production systems in the context of digital factory (Feldmann, Schmuck et al. 2008), the definition of production skills (Pfrommer, Schleipen et al. 2013) and in the orchestration of service-based manufacturing systems (Pfrommer, Stogl et al. 2014, 2015). Moreover, the PPR model is taken as a base in several international standards motivating the International Organization for Standardization (ISO) mandate to adopt it as a management tool for managing modularity (Cutting-Decelle, Young et al. 2007).

Some examples are to be found in the literature referring to the PPR model with respect to industrial robotics: Schraft and Hartfuss (1994) report knowledge-based programming systems for robotic arc welding where geometric information is defined from the CAD model and the welding process information is encoded as rules for generating robot movements. Hartfuss (1996) specifies a knowledge-based method identifying the product spectrum, the welding process, and the robot itself for welding activities in the steel building construction sector. Arikan and Balkan (2000) use segmented surface models from a computer-assisted design software for programming the robotic spray painting using process models.

Furthermore, Mark, Bohlin et al. (2014) describe a system which generates collision-free paths for robotic sealing stations in paint shops by modeling the process and a task definition based on the geometry for optimizing the process sequence. Stenmark and Malec (2015) define the skill of a robot as the link of the three components: product, process, and resource. Defining for the case of a robot-based production system the skill as a "coordination of parameterized actions". Perzylo, Griffiths et al. (2015) relate a preliminary evaluation of an industrial assembly use-case and a wood-working use-case (Perzylo, Somani et al. 2016) in which the domain specific knowledge is modeled with the PPR approach. An error source structure for the identification and model-based compensation of errors in robotic manufacturing has been proposed using the Product, Process, and Resource (PPR) model (Diaz Posada, Schneider et al. 2016).

In addition to the previously mentioned exemplifications, it is worth mentioning that the PPR model is also widely implemented as a semantic base in some state-of-the-art OLP industrial software which includes robotic resources such as Delmia (Delmia 2017). Furthermore, the data format named Automation Markup Language (AML) based on eXtensible Markup Language (XML) supports also the PPR approach. Examples have been proposed for the storage and exchange of plant engineering information (Drath, Lüder et al. 2008). Also, Schleipen and Drath (2009) describe the modeling of manufacturing plants using the three-view-concept with AML. Extension of this work derives skill of production plants from the PPR view (Schleipen, Pfrommer et al. 2014).

On the robotic industrial implementation, the AML data format also reports advantages. Kuhlenkoetter, Schyja et al. (2010) report robot workcell simulation. Krug (2013) utilizes AML for the automatic configuration of robotic systems in a plug and produce approach. Moreover, uses of AML on the Robot Operating System (ROS) are reported by linking model-driven approaches and ontological reasoning for industrial applications (Hua, Zander et al. 2016). As a complement to the AML data format, the PPR manufacturing processes classification has been also embedded into automatic configuration files for industrial processes like the XML format. An example of this is described in the publication presented by Sang, Tae et al. (2010).

For illustrating this concept, Figure 2.3 depicts the PPR model. Figure 2.4 shows the PPR model structure in the assembly task of an industrial robot using the Delmia software. A comprehensive review of knowledge-based robotic manufacturing based on the PPR approach has been presented to classify the different models available in each category (Kuss, Diaz Posada et al. 2016). A detailed review of each category is offered below.

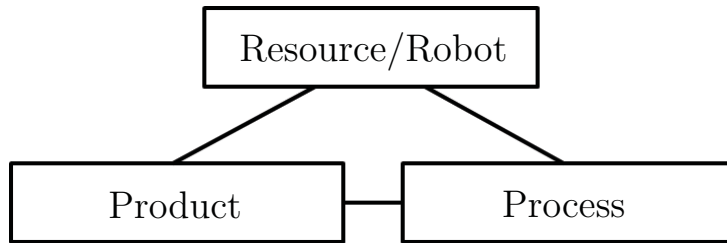


Figure 2.3: The PPR Model.

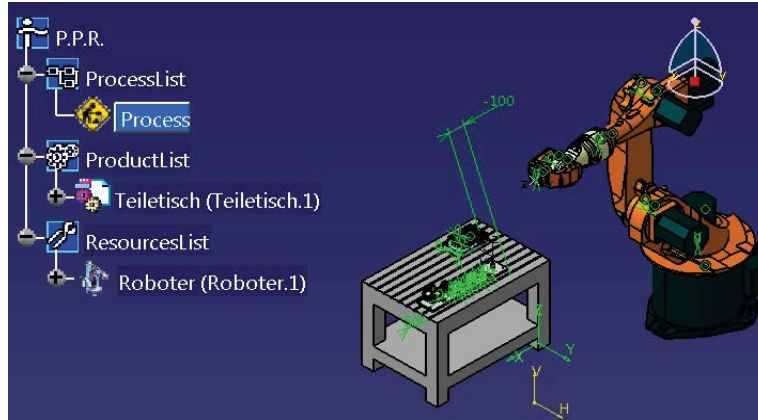


Figure 2.4: PPR model in Delmia for the simulation of a robot task.

2.2.2 The Product Model

As defined by Kuss, Diaz Posada et al. (2016, p.349) “The product models describe the properties and geometries of the manufactured goods at different levels of completion”. These models can be classified as suggested by Nielsen (2003) as follows: structure-oriented, geometry-oriented, feature-oriented, knowledge-based, and integrated product models. Due to the nature of this dissertation, special interest is given to the geometry and feature classification.

Representations of product models have been leveraged thanks to the development of CAD technologies due to the integration of semantic representation of object geometries and features. This representation permits the characterization of products and resources in virtual environments allowing

the design of robot products and cells in software (refer to Section 2.3). CAD models for automated path planning can be classified into parametric and mesh (Chen, Fuhlbrigge et al. 2008). Mesh models tile a surface using other geometric shapes, normally polygons or triangles, for constructing the mesh. Parametric models instead offer “precise representation of the part geometry. It provides entities such as points, lines, arcs, splines, NURBS surfaces, and solid elements” (Chen, Fuhlbrigge et al. 2008, p.523). Several formats are nowadays available for modeling products using CAD software. Some of the most relevant examples of neutral CAD formats (non-proprietary) are: the STandard for the Exchange of Product model data (STEP) (ISO 10303-21), the COLLADA (ISO/PAS 17506), the STL, the VRML, the JT, the IGES, and the Parasolid.

The latest research on the use of CAD formats for the improvement of the industrial programming experience is based on the extrapolation of ontological representations of manufacturing tasks (Perzylo, Somani et al. 2015, Kuss, Ksensow et al. 2017). The description of the manufacturing task allows the linking of geometrical and topological feature descriptions such as lines, curves, circumferences, or splines to the discrete description of the CAD format. These feature descriptions constituted the base of the boundary representation which describes shapes using the limits of the topological and geometrical entities.

Some applications are to be found in the literature for improving robotic programmability based on CAD (Perzylo, Griffiths et al. 2015, Perzylo, Somani et al. 2016, Kuss, Ksensow et al. 2017). All of them are based on the identification of features on the product. As defined by Kuss, Ksensow et al. (2017), there are three main approaches for determining these features: by including them in the design phase, by mapping techniques, and by recognizing the feature which opens the possibilities to OLP systems for including such descriptions.

2.2.3 The Process Model

The definition of process model proposed by Nielsen (2003, p.63) state that “a manufacturing process is a process that transforms a physical object from one state to another”. Two main connotations of process models are to be found in the literature: the semantic and the mathematical.

The first connotation focuses on the set of actions required for manufacturing a product. Some formats have been proposed to specify the process in this category. The Process Specification Language (PSL) offers a set of logical terms standardized under the ISO 18629 for describing processes. Based on the Unified Modeling Language suitable for programming UML/P, the domain specific language LightRocks (Light Weight Robot Coding for Skills) has been proposed for programming robots (Thomas, Hirzinger et al. 2013). The Rule Markup Initiative (RuleML) and the Production Rule Representation (PRR) are global organizations promoting industry standards for specifying process models. The second connotation of process models refers to the mathematical modeling of process characteristics acquired through empirical or theoretical methods. Several models exist for the different manufacturing processes.

2.2.4 The Resource/Robot Model

The resource model is defined by Nielsen (2003, p.66) as “an object that is used to fulfill some needs of the user. Hence, for the purpose of manufacturing, categories of resources include equipment, humans,

and information”. Due to the nature of this research, the resources considered in this dissertation are 6-joint serial industrial robots. This sort of robots will be referred as robot through this text. Mathematical models for these robots have been widely studied, are available, and are firmly established in literature such as the forward kinematics with the Denavit-Hartenberg (DH) parametrization, the inverse kinematics, the robot dynamic modeling (Craig 1989, Siciliano, Sciavicco et al. 2010), and the robot compliance/stiffness models (refer to Subsection 2.5.2). These models will be used as basis for the transformations between Cartesian and Joint space through this study.

General knowledge-representations for the kinematic and dynamic of robots have been proposed as well by some authors. For instance, Djuric and ElMaraghy (2009) present a concept for representing different robotic systems in a reconfigurable plant model. Kuhlenkoetter, Schyja et al. (2010) describe the parametrization of robot models with support of COLLADA (Diankov, Ueda et al. 2011), AML, and MathML for robot workcell simulations. Lately, the Universal Robot Description Format (URDF) has also appeared as a widely used XML-based description format for representing robots in the Robot Operating System (ROS).

This section (2.2) presented the state of the art with respect to the knowledge representation in robotic manufacturing. The Product, Process, and Resource (PPR) model is identified as a widespread model in industry. Therefore, the PPR components with focus on RMPs were also detailed. Due to the previous, the PPR model is adopted in this research as base for the description of the RMPs. Arguments for this adoption are further detailed in Subsection 2.7.1.

2.3 Offline Programming Software for Robotic Manufacturing Processes

Several commercial and open-source software are available today in the market for the simulation and generation of paths for industrial robotic systems. This section introduces some relevant software in this area. Furthermore, Table 2.3 summarizes information of some of these software packages. The information of the table has been extracted from the official web-pages and communication with the software providers before the month of June in 2017.

Nowadays, the geometric modeling and the features definition functionalities are state-of-the-art characteristics of OLP software. Simulations of robotic paths and collision checking algorithms are also common functions. Specialized robot packages offer analysis of reachability, singularities, joint limits, and wrist flips which are fields that have been intensively studied and further decoded into software. Desired clearance to obstacles can be also defined in some applications in order to condition the manual path generation. Some solutions, just commercial ones, offer specific packages and functionalities for manufacturing processes. Those functions include mainly specification of manufacturing tools and time optimization of devices sequences for processes such as drilling and riveting.

Other OLP software include planning for sensor compensated process, an example of this is the ABB Robot studio software. As another functionality, the simulation of robot signals is provided by software packages such as Gazebo, V-Rep, and RobCAM Siemens. Even though there is an important development of the OLP software for robotic manufacturing, it is not observed that robot movements are automatically optimized in relation to process and robot optimization criteria. The analysis tools,

in the best cases, provide the end-user with a Graphic User Interface (GUI) in order to manipulate and adapt the robot movements in case of infeasible states such as collision, joint limits, or singularities. These manual adaptations explain the requirement of specific robotic and manufacturing process knowledge in order to accomplish the programming task.

Table 2.3: Some relevant OLP software for the generation of robotic paths for manufacturing processes including open-source packages.

Characteristics/ Software Packages	Commercial or Open-Source	Robotic Analysis Tools	Manufacturing Process Tools	Main Characteristics
Robotmaster (MasterCAM)	commercial	- collision checking - reachability - singularities - joint limits - robot cycle time - constant tool orientation	- sequencing strategy - no process simulation - integration of specific process parameters	- all CAD formats - configuration of tooling axis - macros available for control of peripherals - flexible programming through SDK (C/C++)
KineoCAM Siemens Originally Kineo (LAAS-CNRS)	commercial	- collision checking		- desired clearance to obstacles and length definition
Gazebo	open-source	- collision checking - include dynamic simulation engines - different sort of robots	n/a	- generates sensor data and noise
Delfoi - Visual components	commercial	- collision checking - reachability - singularities - joint limits	- Delfoi ARC offers (Welding Procedure Specification) - Delfoi CUT offers creations of milling tools to be included in the robot simulation	- all CAD formats - flexible programming through Python - possible finite elements functionalities - configuration of tooling axis
V-Rep	open-source	- include robot libraries	n/a	- minimum distance calculator - include sensor (Proximity and vision) - interface with the reflexxes library
CGTech - Vericut VDAF	commercial	- collision checking - singularities - processing time	- sequencing strategy - visualization process steps	- all CAD formats - adaption through internal post-processor - mathematical formulas and equations possible - configuration of tooling axis
CENIT	commercial	- collision checking	- sequencing strategy and categorization - definition of process parameters. e.g. spray pistols setups	- all CAD formats - based on Delmia V5 - configuration of peripherals possible - configuration of tooling axis

Characteristics /Software Packages	Commercial/ Open-Source	Robotic Analysis Tools	Manufacturing Process Tools	Main Characteristics
ABB Robot studio - Machining power pack	commercial	- collision checking - singularity - checking of robotic path for collision, singularity, wrist flips	- set of machining process parameters - pre-defined path generation strategies and patterns	- planning for force and torque sensor compensated processes - intuitive programming for end-user of force and torque control
Delmia	commercial	- cycle time optimization - auto-positioning of robots - Plotting of the workspace Envelope for design purposes	- Special toolboxes for milling, welding, polishing and painting - Setting of process parameters	- Open system which can be program based on XML files - Virtual teach pendant for interactive robot programming

In addition to the previously presented information, some surveys and comparisons are found in the literature with respect to OLP for robotic systems. These studies are mainly focused on open-source tools as stated by some authors due to its easy accessibility. For instance, Staranowicz and Mariottini (2011), Tom, Yuval et al. (2015) present an overview of eight and three open-source and commercial software respectively used in robotics. Ivaldi, Padois et al. (2014) show results of a comparative study based on end-user feedback for eighteen open-source and commercial robot simulations tools. Erez, Tassa et al. (2015) compare the robotic simulation tools Bullet, Havok, MuJoCo, ODE and PhysX.

In comparison to the open-source simulators, few surveys are found in the peer-review literature for commercial solutions. One of them is presented by Klingstam and Gullander (1999) in which an overview of some of the simulation tools for computer-aided production engineering is offered. Some robotic applications are reviewed and characteristics such as the supported robot languages are evaluated. Wang, Li et al. (2011) describe an architecture for the development of commercial robot simulation systems.

Even though few surveys are available in the literature for the commercial products, several authors report the implementation of robotic use-cases using them. Pires, Godinho et al. (2004) present a welding robotic systems program in which the task definition of the process is facilitated by the use of the CAD software AUTOCAD. An extension of this approach also for programming welding robots based on CAD has been presented with Autodesk Inventor (Neto, Mendes et al. 2013, Neto and Mendes 2013). Chen, Fuhlbrigge et al. (2008) uses also CAD information for automating the spray painting with industrial robots based on tessellated CAD models.

Other authors, such Mohammad, Babariya et al. (2008), detail the modeling of a deburring process using Delmia. Vosniakos and Chronopoulos (2009) describe a general-purpose CAD turned into a robot programming and simulation environment exemplifying it into a loading/unloading task. Polden, Pan et al. (2011) report an offline programming of a welding cell using Delmia by automating some of the tasks such as the semi-automatic method of defining weld seams and the geometric definition of frames over this tasks. Stenmark and Malec (2015) describe an implementation of a generic knowledge-base system architecture using ABB Robot studio for realizing force-controlled tasks performed by one-armed and two-armed robots. Behnam, Berbyuk et al. (2010) implement in ABB Robot Studio an optimization

of the robot and task positioning. Doan, Tao et al. (2016) use also ABB Robot Studio for optimizing the redundancy in robotic welding using a particle swarm approach. Ahmed, Yuan et al. (2015) plan multiple layers for robotic welding in Matlab with a simulation in the RobotMaster environment.

This section presented the advances on robotic OLP software. The next three sections describe the state of the art for the path generation of the robotic manufacturing processes and use-cases studied in this thesis. Sections 2.4, 2.5 and 2.6 present the state of the art of the implemented use-cases: the optimized collision avoidance in robotic welding, the optimized stiffness for robotic milling, and the optimized sensing strategy for sensor-guided robotic deburring, respectively. Based on the state of the art of each use-case, the problematic and motivation for each of these, with respect to the topic, is presented in Subsection 2.7.2.

2.4 Path Generation in Robotic Welding

The collision between the robot, welding workpieces or products, and additional objects is one of the most performed and studied task in robotic manufacturing. This is one of the major problems which programmers face during robot programming of complex welding workpieces. Moreover, the collision avoidance is a topic of high interest in the robotic community due to the intrinsic nature of the robot being a movable object in space without the robotic welding process being an exception. The following subsection provides an overview of the path generation for the robotic welding process.

2.4.1 Methods and Algorithms in Robotic Welding

Several path planning methods and algorithms (refer to Section 2.1) have been used to the robotic welding task. The grid-based planning have been reported by Verbarg (1997, 1998) where a non-optimal algorithm for finding collision-free paths for the welding robots in Audi AG in Ingolstadt was implemented. For this application, the A* algorithm was implemented in synergy of an heuristic for controlling the expansions of the algorithm. Extending to sample-based planning, Larkin, Short et al. (2016) generate paths for robotic welding from STL models by decomposing the problem into sequential sub-problems. Tags are generated from the STL files, A* algorithms are used to find welding motions, then a PRM algorithm is implemented for transition motions. Erdős, Kemény et al. (2013), Polden (2014) present other examples of grid- and sample-based approaches reviewed in the next subsections. Some other publications demonstrate the implementation of grid-based methods such as A* and potential fields in Subsection 2.4.3.

Another programming method also reported on welding scenarios is the Programming by Demonstration. Helms (2007), Meyer (2011) report implementation of the PbD approach for welding robots (For more detail in this application refer to Subsection 2.1.1). Furthermore, geometrical and mathematical based path planning have been also reported in some publications. As example, Chou, You et al. (2007) compute joint trajectories over a reconstructed surface modeled with a B-spline of third order by commanding the welding torch over points on the surface taking into account its normal vector for the orientation. Lima and Bracarense (2009) present a geometrical planning for shielded metal arc welding. Other mathematical approaches such as the presented by Huo and Baron (2008, 2005), Bayan, Pereira et al. (2005) are reference below in Subsection 2.4.4.

As an example of sensor-based programming Kim, Rhee et al. (1999) use information of a laser vision sensor for programming robotic welding tasks. The approach of using sensors is in some occasions implemented in practice. Kah, Shrestha et al. (2015) reference some applications of sensor-based programming for robotic welding. Optimal control approaches are also to be found in the literature (Gerdtts, Henrion et al. 2012, Björkenstam, Spensieri et al. 2014, Landry, Gerdtts et al. 2014). These references are reviewed later in this section. Similarly references are reviewed later for bio-inspired algorithms (Mitsi, Bouzakis et al. 2008, Liao, Wang et al. 2010, Wang, Shi et al. 2015, 2016, Doan, Tao et al. 2016). The previous demonstrates the wide spectrum of methods and algorithms used for the robotic welding path planning problem.

2.4.2 Knowledge-Based Approach and OLP in Robotic Welding

The area of knowledge-based programming (refer to Section 2.2) and CAD-based path generation have been driven by the robotic welding on the manufacturing sphere. Early definitions of knowledge-based welding are reported by Schraft and Hartfuss (1994) by describing a knowledge-based programming for robotic arc welding which generates programs from CAD data by defining the following knowledge bases: weld feature interpretation, minimization of cycle time, and generation of robot movements. More recent applications such as the one of Bickendorf (2014) proposes an automatic CAD-based offline-programming based on technological knowledge and rules described in XML formats. Computing of geometrical paths for difficult reachable weld seams for the robotic welding of ship-sub-assemblies with seam-tracking sensors are programmed based on these rules. Another example is the one offered by Ding, Pan et al. (2015a,b) where a process planning strategy for arc additive manufacturing by decomposing and sliding the STL model of the product and using tool-path patterns is performed using different programming methods and algorithms.

2.4.3 Collision Avoidance in Robotic Welding

The collision avoidance has been studied in the robotic community from the early beginning. The basic function for approaching collision avoidance in OLP is the collision detection. Several algorithms and methods have been proposed (Kockara, Halic et al. 2007, Jiménez, Thomas et al. 2001). Moreover, nowadays open-source packages offer some algorithms such the case of the Faster Triangle-Triangle Intersection Tests (Devillers and Guigue 2002). Lozano-Perez and Wesley (1979), who introduced also the C-space, describe an algorithm for planning collision-free paths for polyhedral objects among polyhedral obstacles. Implementations on industrial scenarios have been also proposed. For instance, Hein, Salonia et al. (2001) present a collision-free time optimized algorithm based on rounding for industrial environments. Likewise, Ting, Lei et al. (2002) report collision avoidance algorithms for industrial robots by means of using the wave expansion methods and the depth-first search approach. Gerdtts, Henrion et al. (2012) instead, describe a time-optimal control problem to find the fastest trajectory for a robot while performing collision avoidance.

Methods such as the potential fields have been also reported in the literature for avoiding collisions. Juang (1998) describes the online collision avoidance for industrial robots by following the negative gradient of the potential field method. Similarly, Csiszar, Drust et al. (2012) present an online

path planning collision avoidance algorithm for industrial robots using the artificial potential field method.

Other robotic scenarios have also arisen concerning the collision avoidance. Multiple robot sequencing has called also the attention of researchers with respect to the collision avoidance. Examples are offered by Larsen, Pham et al. (2015) who present a collision-free path planning for cooperating robots in an industrial environment and Spensieri, Carlson et al. (2016b) who describe an iterative approach for planning collision-free routing and scheduling on multiple robot stations in car manufacturing by first using an exact branch and bound method and then solving the collision problems.

The welding process has not been an exception in the collision avoidance research. Landry, Gerdtts et al. (2014) presents a collision-free path planning of welding robots by solving an optimal control problem representing the objects by convex polyhedra and using the sequential quadratic method. Ahmed, Yuan et al. (2015) present a collision-free path planning for multiple layer welding of offshore oil rigs using the A* algorithm by building a weld bead layout based on the cross-sectional area of the joint. Polden (2014), Polden, Pan et al. (2017) test an automatic OLP system in an industrial robotic welding system with 13 DoF by using an extension of the Lazy PRM and post-planning optimization for the robot transitions and auxiliary motions which are computed collision-free. Mitsi, Bouzakis et al. (2008) report a time optimized collision-free path planning determining an amount of intermediate configurations as node poses for robotic welding by using a hybrid method combining a genetic algorithm, a hill-climbing method, and a constraint handling method.

2.4.4 Path Optimization in Robotic Welding

In the field of welding, the need of optimization of controllable parameters and an approach for modeling the process, robot, and sensors for preparing the task offline has been pointed out by Bolmsjö, Olsson et al. (2002). Stand-alone optimization of the robot motion with respect to the manufacturing process are found in the literature. For instance, Huo and Baron (2008) studied a joint-limits avoidance strategy by a virtual rotation around the electrode axis for arc-welding robots implemented in joint space. Dolgui and Pashkevich (2004), Dolgui, Pashkevich et al. (2005) proposed that the weld may be processed in the out-of-position location charged by reduction of the welding speed in order to satisfy the quality specifications and considered a cluster-based welding operations plan for minimizing processing time in an arc-welding robotic cell with positioning table. In this work, a geometric description of the feature is defined and the welding torch spatial location is described relative to the welding frame.

Furthermore, Huo and Baron (2005) propose using the twist of the end-effector decomposing it into two orthogonal sub-spaces in order to solve functional redundant tasks. Similarly, the same authors (Huo and Baron 2008) introduce a redundancy-resolution algorithm to optimize the joint space trajectory of an arc-welding task for avoiding joint-limits and singularities. Bayan, Pereira et al. (2005) proposed using the screw theory for optimal welding positioning and trajectory planning using also virtual chains providing bi-dimensional examples. Ericsson and Nylén (2007) describe the use of robotic simulation and finite elements simulation for minimizing distortion while keeping complete joint penetration by optimizing the robot welding speed.

The sequencing and optimization of transition motions report several studies in the literature. As example, Behnam, Berbyuk et al. (2010) present a proposed algorithm for optimizing the cycle time

by changing the path position with respect to the robot position in simulation environments. Erdős, Kemény et al. (2013) relate the computation of a close-to-optimal sequence of welding tasks for the remote laser welding process based on a solution derived from the traveling salesman problem and path planning heuristics. Carlson, Spensieri et al. (2014) use systematic search algorithms for minimizing dimensional variation and robot movement required time in welding stations for the automotive car body assembly process.

More on the sequencing optimization is presented by Björkenstam, Spensieri et al. (2014). These authors describe an optimal sequencing of industrial robots using optimal control for the welding of 20 stud welding points of car body using an industrial robot. Spensieri, Carlson et al. (2016a) describe the optimization of the robot placement for improving cycle time without taking into account the orientations by having as input the robot model, the CAD model of the workpiece, and a set of tasks. Several OLP systems offer packages for optimizing robot sequences (refer to Section 2.3).

Further optimization is also found in the literature by using bio-inspired algorithms (refer to Subsection 2.1.1). In this area, Liao, Wang et al. (2010) relate a time-optimal trajectory planning for a 6 joint welding robot using adaptive genetic algorithms between the points of discretized trajectory. Wang, Shi et al. (2015) optimize the robotic spot welding sequences with a partition mutation. Wang, Shi et al. (2016) use a double global optimum genetic algorithm particle swarm optimization for optimizing paths for the spot welding process by taking into account the shortest path. Doan, Tao et al. (2016) use a modified particle swarm for optimizing the energy consumption of a welding robot by exploring one rotational DoF.

On the workpiece position optimization for the robotic welding process, some works are also to be found. An energy reduction consumption approach for an exemplary spot welding task optimizing the robot platform position by varying its position in a discrete grid has been investigated (Eggers, Hansen et al. 2014). A workflow has been outlined in which a workpiece is placed taking into account the defined task points of a path avoiding collision without including a fine placement optimization for a remote robotic laser welding process (Erdős, Kardos et al. 2016). The optimization of the robot base position has been discussed for reaching the predefined tasks and for minimizing cycle time using a derivative-free optimization algorithm taking into account collisions for a spot welding application (Spensieri, Carlson et al. 2016a).

2.5 Path Generation for Improving Robotic Milling Accuracy

In the current industrial scenario, robots are rarely used in contact operations such as machining and finishing as they entail unsatisfactory accuracies, complex programming, and control methods. Further, the disparity between accuracy specifications, communication technologies, and control methods required for such operations, calls for greater efforts in robot programming and control.

Evidence of these challenges is that industrial robots are still not widely used for machining purposes. Proof of that are the IFR statistics. According to the IFR, just 1 %, 1.4 %, 1.7 % and 1.1 % of the operational stock of industrial robots at the end of each year, respectively 2014, 2015, 2016, and 2017 were related to processing tasks such as mechanical cutting, grinding, deburring, milling or polishing (IFR Statistical Department 2014, 2015, 2016, 2017), which is a small quantity taking into account the flexibility and reconfigurability of robotic systems.

One of the reasons for this low stock of processing robots is the inaccuracy. A Computerized Numerical Control (CNC) is 50 times stiffer than a robot (Tyapin, Kaldestad et al. 2016) and the robot is by factor 100 more inaccurate than machine tools ($\pm 1 \mu\text{m}$ in comparison with $\pm 0.1 \text{ mm}$) (Atmosudiro, Keinert et al. 2014). Another important reason is the complex program generation process and the difficulty of optimizing robotic paths for improving quality. Robots in processing possess advantages over conventional tool machines when comparing dexterity overextended workspaces and the changeability to new processes and products (Jayaweera and Webb 2011) but are not adopted due to the many challenges faced.

Reasons for addressing optimization and automatic programming are that several advantages can be derived if robots were widely used for machining operations. Some of these reasons are: (1) less work space will be required if robots are used for machining instead of CNC machines (Atmosudiro, Keinert et al. 2014, Slavkovic, Milutinovic et al. 2014); (2) robots are now less expensive than CNC machines and could represent cost savings in industry (Minhas, Lehmann et al. 2011, Pandremenos, Doukas et al. 2011); (3) robots are more flexible and can also be used for doing other tasks in addition to the machining operation, such as picking the workpiece, if the robot is provided with a tool changer and a gripper, increasing the productivity (Karim and Verl 2013); (4) the dexterity of the robot can be used more effectively to machine complex machining parts (Atmosudiro, Keinert et al. 2014, Pandremenos, Doukas et al. 2011); and furthermore, (5) a robot can substitute heavy manual work, which may be a risk for human health (Jayaweera and Webb 2011).

Several authors have also pointed out challenges and reasons for addressing automatic programming and compensation in this regard. Three main obstacles, (1) insufficient rigidity, (2) poor accuracy and (3) complex programming, have been identified for the machining of hard materials with robotic systems (Karim and Verl 2013). The demand for quick reconfiguration of robots for precise machining applications has been pointed out as a required strategy for responding to the current demand of product individualization (Minhas, Lehmann et al. 2011). Similarly, the complexity of programming has been pointed out as one of the most important limiting factors (Slavkovic, Milutinovic et al. 2014).

The need of a path programming module embedded in Computer-Aided Manufacturing (CAM) software for compensating machining errors such as chatter, produced due to low robot stiffness and high machining forces, and improving quality while taking into account unreachabilities and singularities has been also pointed out as a need for the offline programming in robot high-accuracy machining (Leali, Pini et al. 2013, Pandremenos, Doukas et al. 2011). Robot stiffness, vibrations, and system natural frequencies are also referred as critical factors (Iglesias, Sebastián et al. 2015).

In reaction to these challenges on robotic machining, the scientific community has proposed error source modeling identification, compensation, and some optimization. On the error source identification, for instance, Schneider, Ansaloni et al. (2013) have experimentally determined several error sources and proposed strategies for improving robotic machining accuracy through modular compensation (Schneider, Drust et al. 2014, Schneider 2017). Following subsections give further insights on the compensation (2.5.1) and optimization (2.5.3) in this ambit.

2.5.1 Error Compensation in Robotic Milling

The error source compensation in robotic milling can be classified as model-based and sensor-based compensation (Schneider, Drust et al. 2014). Both approaches are presented for giving the reader an insight of the compensation techniques and modeled knowledge used for improving accuracy. In both approaches, it is noticed that milling process models are widely used. Several investigations exist in this regard outside the robotic community in form of standards or process specifications (Altintas 2000, Budak and Tekeli 2005, Gomeringer 2014). Some of them have been modeled also in relation with industrial robots systems (Matsuoka, Shimizu et al. 1999). In this regard, the stiffness/compliance of the robot is widely used in compensation approaches. Subsection 2.5.2 present an overview of these models.

Model-Based Compensation in Robotic Milling

Some authors report approaches and implementations using model-based compensation. Abele, Bauer et al. (2010) predict in simulation the tool displacement for the milling process and propose an offline compensation approach based on the process force and chip geometry calculation, a cutting force model, and a robot model. Previous modeling of the industrial robotic milling process was developed by these authors (Abele, Rothenbücher et al. 2008).

Bauer (2011) implements a sensor-based and a model-based compensation for the robot milling process. The model-based compensation uses a robot dynamic model and a milling force model with which the robot deviations due to process forces are computed and compensated reporting 37 % of error reduction. In this publication, the CAD/CAM software Unigraphics NX 6 is used to generate initial milling trajectories for a specific workpiece. The robot poses and the technological parameters such as feed-rate, spindle speed, and tool are parsed also from this software. The compensation is performed afterwards based on these parameters.

Besides, Lehmann, Halbauer et al. (2012, 2013) propose a three-step approach; machining strategy generation, offline force compensation, and online force compensation; as a strategy to compensate the force-induced accuracy influences. Bauer, Friedmann et al. (2013) experiment with two model-based compensations: The ideal milling force and the deviation mirroring using robot structure models and process force calculation. Haage, Halbauer et al. (2014) present a compensation method for increasing robotic machining accuracy using offline compensation computed based on robot kinematics, a process force and a dynamic robot model used to calculate the compensations embedded into the PowerMill tool. Results demonstrate more prominent deviations on the corners of a 2D machined rhombus with maintained orientation, without using the DoF of the process, during machining (Haage, Halbauer et al. 2014).

Other authors such as Klimchik, Bondarenko et al. (2014) compensate the path by modeling the interaction between the milling tool and the workpiece and experiment in aluminum reporting improvements on the avoidance of static deviation of more than 90 %. Slavkovic, Milutinovic et al. (2014) model cutting forces and the robot compliance in order to predict the cutting force-induced tooltip displacements for offline compensation of cutting force-induced errors in five-axis machining. Experimental results in aluminum depict improvements of 180 μm .

In other researches, opportunities and challenges of productivity increase while planning robotic paths in joint space in an offline approach have also been reported. Experimentation in joint space depicted a higher robot TCP velocity, at expenses of path accuracy, in comparison with the planning in Cartesian space (Atmosudiro, Keinert et al. 2014). Tyapin, Kaldestad et al. (2016) implement an offline path compensation by computing the deformations using estimated process forces based on a static force model and the identified joint stiffness models.

Sensor-Based Compensation in Robotic Milling

Some authors have reported also advances on compensation for robotic milling through sensor-based approaches. Liu, Cheng et al. (2001) propose an adaptive control constraint of machining processes with a constant cutting force control system. Pan and Zhang (2008), Wang, Zhang et al. (2009) introduce an online deformation compensation approach based on a robot structure model for controlling the material removal rate. This approach controls the machining force by adjusting the robot feed speed in real-time for improving the part quality. Bauer (2011) proposes a compensation by comparing camera information and CAD information and using the Iterative Closest Point (ICP) algorithm for compensating product errors in robotic milling.

Furthermore, Sörnmo, Olofsson et al. (2012a) ensure time-efficiency and improve the accuracy of robotic machining processes removing the maximum amount of material per time unit through adaptively controlling the applied force, as well as continuously compensating for the path deviations caused by process forces. In a further work, a control with a piezo-actuated high-dynamic micro manipulator is performed (Sörnmo, Olofsson et al. 2012b). Schneider, Momeni-K et al. (2014) model the robot stiffness and compensate in real-time computed deviations due to forces and torques measured at the robot TCP.

Other controls without compliance or process models, such as the position control of industrial robots in real-time using optical measurement systems for machining purposes have been proposed for improving accuracy (Schneider, Diaz Posada et al. 2013). Schneider (2017) presents also a dual time real-time control compensating the low frequencies with the robot and the high frequencies with an external compensation mechanism in which the spindle was mounted achieving a mean absolute deviation of $32\text{ }\mu\text{m}$ for the milling of steel.

2.5.2 Compliance Models for Robotic Milling

As described in the previous subsections (model- and sensor-based compensation), several authors refer stiffness/compliance models for the compensation in robotic milling. Different models and methods found in the literature are referenced below: One of them calculates the Cartesian stiffness based on the polar stiffness and the use of the Jacobian matrix (Abele, Weigold et al. 2007). The same authors propose later the use of virtual joints (Abele, Bauer et al. 2008). The identification of the diagonal joint stiffness matrix is approached by applying external wrenches and the modeling of a diagonal joint stiffness matrix (Dumas, Caro et al. 2010, 2011).

Moreover, the stiffness modeling of heavy industrial robots with gravity compensation has also been studied (Klimchik, Wu et al. 2013). Another method for measuring gear deviations uses a laser interferometer-based technique (Slamani and Bonev 2013). The robot stiffness is also identified by

the clamping method in which the robot is locked in a closed kinematic loop and then exercises joint-based movements while sensing motor position and torque (Haage, Halbauer et al. 2014). Schneider, Momeni-K et al. (2014) identify robot stiffness as linear descriptions on the ranges where no backlash is observed and as non-linear functions when backlash is noticed. Third order polynomials are addressed when gear deformation is noticed. Cho, Kim et al. (2017) use deflected circular trajectories for the identification of kinematic screw and joint compliance.

2.5.3 Path Optimization in Robotic Milling

Several authors have pointed out the optimization as an enabler for improving accuracy and programmability for robotic milling. Optimization of the parameters, such as workpiece location, natural frequencies of the robot, and robot structure in given configurations, are pointed out as important criteria for the machining process (Lehmann, Halbauer et al. 2012). The mapping of a rigidity map for choosing optimal joint configurations has been stated as one of the future research issues (Chen and Dong 2013). Bauer (2011) points out in the outlook of his thesis that multiple criteria optimization for the motion generation is conceivable in particular when errors should be compensated in rounding areas such as corners. In this work, it is mentioned that this could reduce programming times and increase accuracy. Lehmann, Halbauer et al. (2012) identify machining parameters and characteristics such as the location of the workpiece in the robot workspace, appropriate feed rates, number of points considered for the motion planner, axes reversals, and cut strategies as optimizers to improve machining quality. A look-ahead function for improving accuracy is proposed in this paper as future work.

For improving robotic machining it has also been determined that the combination of holistic programming in CAM and the optimization of the redundancy can be used in conjunction with kinematic parameter optimization (Schneider, Diaz Posada et al. 2014). Karim, Hitzer et al. (2017) also state, after analyzing the dynamic behavior of a six-axis industrial robot within the entire workspace in respect of machining tasks, that the optimal pose in consideration of the excitation and oscillation directions can be chosen on the basis of characterization.

Some stand-alone optimizations are to be found in the literature. Most of them are solved through numerical or optimal control approaches. A numerical approach is presented by Makhanov, Batanov et al. (2002) in which global interpolation of the required surface by a virtual surface is used to optimize robotic milling taking into account the scallop height. Owen, Croft et al. (2008) address an offline planner for optimizing stiffness through trajectory resolution finding initially a stiffer position and providing several input configurations for later using a real-time trajectory planner with tool speed control for reducing demand of actual torques. Matsas and Vosniakos (2010) use two genetic algorithms, one for maximizing the manipulability of the robot and the second one for minimizing joint torque loads, in robotic milling.

Furthermore Xiao and Huan (2012), relate a numerical optimization approach to solve the redundancy of the milling process. An optimization problem is proposed and the measure of manipulability is used for resolving wrist singularities. This problem is optimized through discrete cut points. Subrin, Sabourin et al. (2013) compare two different kinematics, a hybrid robot with a parallelogram close loop by defining an optimization problem and the criteria of stiffness, dexterity in a given direction, and mechanical performance. Nelaturi, Burton et al. (2015) describe a methodology by defining a

configuration space obstacle for planning automatically time/cost near-optimal plans for the machining of products. In contrast to the last three presented optimizations, this one does not use optimal control and numerical techniques, this planning process is solved using a weighted A* search. Xiong, Ding et al. (2017) propose a feed-direction stiffness based index in order to optimize each cutting location point.

The automatic robot stiffness pose optimization for robotic processes has been presented in literature. The redundancy of the tool tip is explored for determining the minimum deviation in the z direction in Cartesian space due to the end-effector and joints weights using the compliance model of a Kuka robot modeled in joint space. 7th order polynomials are used to fit deformations for finding the most rigid pose. Figure 2.5a depicts the redundancy of the process. Figure 2.5b describes the fitted polynomial for optimizing robot stiffness. Moreover, different criteria are defined, such as end-effector stiffness optimization in the process direction, damping, collision, and reachability. Additionally, different interpretations for applications, such as drilling, deburring, cutting, and milling are described (Schneider, Diaz Posada et al. 2015). Mousavi, Gagnol et al. (2017) present a regenerative chatter vibration block for assuring that the robot keeps its path through the stable milling zone by using the functional resolution of the robot. The block is based on a dynamic model of the robot and a cutting force model.

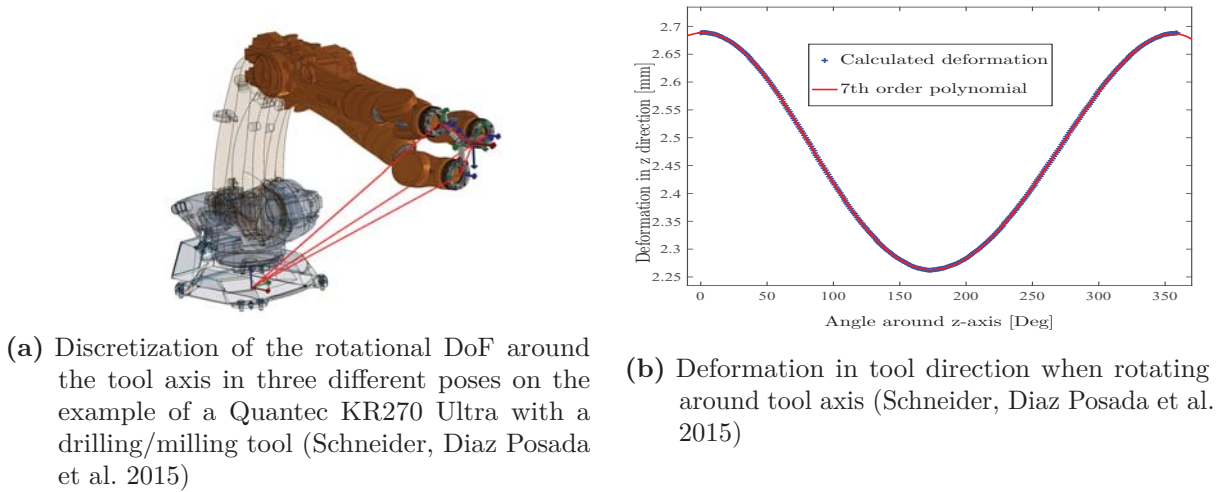


Figure 2.5: Robot stiffness pose optimization.

Workpiece placement optimization for milling have also been described in the literature. The dependency of the robot stiffness with respect to the workpiece position is the main optimized criterion. Genetic algorithms have been proposed for optimizing the positioning of the workpiece in respect to the power consumption of a Stewart platform (Lopes and Pires 2011, Pires, Machado et al. 2014). The best placement of the workpiece have been optimized taking into account the elastostatic robot and the cutting force models using genetic algorithms (Caro, Dumas et al. 2013). Dumas, Caro et al. (2011) describes a workpiece positioning optimization problem using the kinematic redundancy of the milling process. For solving this problem, the constraints and the optimization are defined and solved with required human intervention using the MasterCAM software for two different workpiece positions predefined in the workspace.

2.6 Path Generation for Error Compensation in Robotic Deburring

Robots are low-cost, reliable and flexible manufacturing solutions for industrial automation and therefore they could be extensively used where the lot size is small and the variation of profile geometries is high such as the deburring process. However, like the milling process, they are not widely used in industry due to their complex programming, process configuration, and required accuracy (Jayaweera and Webb 2011). In a contact operation such as deburring, in order to compensate errors due to robotic interaction with the tool and the workpiece for facilitating the programming experience, one can use either offline or online compensation/programming strategies or a combination of both (Dietz, Schneider et al. 2012).

While offline compensation/programming requires an accurate process model, the online method requires integrating sensor systems to measure errors in real-time. Offline compensation/programming methods use a combination of robot model and a machining process model to modify the robot tool path offline for machining. On the other hand, online methods compensate the robotic path errors or interaction forces during the process, based on whether position or force is measured. An overview of implementations and approaches for both methodologies, which serve as the basis for the development of the deburring use-case presented in this dissertation are offered below.

2.6.1 Offline Compensation and Programming in Robotic Deburring

Offline compensation/programming like in the previously described processes is also based on OLP systems. Some authors have reported programming using these software tools. As examples, Mohammad, Babariya et al. (2008) present an offline simulation of the robotic deburring process and the required configuration steps using an OLP software. Leali, Pellicciari et al. (2013) describe an offline programming method for the robotic deburring of aerospace components enhanced with digitalization techniques for localizing the workpiece and calibrating the robot. Vuong, Lim et al. (2013) review the simulation and offline programming for contact operations and present a programming case study for the automated surface grinding.

In addition to the programming based on simulation environments, sensing systems are used in synergy for enhancing the offline programmability and the accuracy of deburring systems. Sensors are used before the execution of the process to acquire information of the real scene. Several authors report the development of this kind of programming and compensation. Examples are: Lee, Kang et al. (1999) who implement a burr shape recognition based automatic robotic deburring system with laser vision sensors in which the burr is modeled, reconstructed, separated and matched in order to adapt a pass path planner. Jayaweera and Webb (2010, 2011) who use a laser scanner sensor to determine the component's exact location prior the deburring operation of aero-engine components.

Moreover, Leo and Selvaraj (2014) who present a five axis robotic system provided with an external optic camera used for recognizing workpiece positions and edges for generating robot programs. Kuss, Drust et al. (2016) adapt offline generated paths to shape deviations of the workpieces for the robotic deburring processes by describing dimensional tolerance specifications and matching the CAD model and a point cloud taken with a 2D laser scanner of the workpiece using the ICP method. Abele, Haehn et al. (2016) describe a time-optimal tool path sequence generation for a robotic deburring process

based on the STL format of the workpiece. In this publication, an A* algorithm is used in order to optimize the time of the deburring sequence.

2.6.2 Online Compensation and Programming in Robotic Deburring

Online compensation and programming are also found in the literature for the robotic deburring use-case. A relevant approach due to the contact nature of the task is to compensate force and torques by sensing them at the robot TCP. Vision systems are also used to detect the workpiece and to react on uncertainties such as the workpiece positioning and tolerances. Combination of both sensing systems (force/torque and vision) are also reported. Moreover, radial and axial compliant spindles usually characterized by pneumatic actuation are also used to partially compensate inaccuracies with the disadvantage of requiring the identification of optimized parameters for achieving good quality deburring (Berselli, Pellicciari et al. 2016).

Some examples of the online deburring are the works from Chung and Kim (2006) who modeled the deburring robotic process and proposed a control design presenting simulation studies with different tools. Song and Song (2013) describe a robotic system which saves a set of manually taught poses over the workpiece, recalculates the position and orientation of the workpiece using the ICP algorithm, and later controls the robot using force control based on an impedance strategy. And Jonsson (2013) who relate a force control application for the deburring/cleaning of iron castings.

Examples of force/torque and vision compensation are described by Bone and Elbestawi (1992) who outline a robotic edge deburring system provided with both force and vision sensors. In this application, the path is generated by a task planner and then later by the action of a controller, the depth is compensated to improve quality. Zhang, Chen et al. (2006) generate online paths based on the feedback of a camera and a force/torque sensor. A hybrid force/vision controller is proposed for the automatic robotic deburring of cast aluminum wheels.

Cheah, Hou et al. (2010) present a visual motion approach with force tracking control taking into account uncertainties of a constraint surface, robot model, and a camera model. Simulation results for planar and curved surfaces are presented in this work. Zhang, Cabaravdic et al. (2004) describe a real-time simulation approach by implementing a neural-network of the overall process model consisting of a contact, force, process, and workpiece model for the grinding of sculptured surfaces.

2.7 Proposition

As presented in the "Problem Statement" section (1.2), the need for an automatic, optimal, and intuitive robotic programming is still a challenge in the programming of manufacturing processes. The previous sections in this chapter introduced four main knowledge areas of the robotic manufacturing, i.e. (1) The robotic path generation, (2) the offline programming software, (3) the knowledge representation for RMPs, and (4) the path generation in the use-cases evaluated in this contribution. This section performs an analysis of them and proposes a synergy among the knowledge areas. This synergy is the base to define the proposed approach and architecture in the next chapter.

As part of the analysis comparisons were conducted among the methods and algorithms as presented in Subsections 2.1.2 and 2.1.5, respectively. The arguments for selecting the OLP and the sample-based techniques for solving the stated problem were given. Subsection 2.7.1 presents as proposition, based on the performed analysis and the state of the art, the potential of the synergy among the four presented knowledge areas for approaching automatic optimal path generation for robotic manufacturing processes. Finally, Subsection 2.7.2 provides the problematic and motivation of the studied use-cases, as proposition for further improvements in these processes.

2.7.1 Synergy Among State-of-the-Art Research Areas

A conclusion of the state-of-the-art analysis is that important advances have been achieved in each investigated field, namely: path generation, knowledge representation, OLP systems, and in the modeling of specific robotic manufacturing process itself. Even though these developments, an automatic, changeable, and structured approach is still missing for overcoming the programming and re-configuration problems of the path generation for RMPs, which is the main contribution of this research. It is observed that the potential of establishing defined synergies, proper coaction, and further research on the presented knowledge areas is a key issue for leading to automatic optimized path generation systems for RMPs in a generalized way. This subsection describes this synergy.

Important advances have been achieved regarding the programming methods and algorithms for RMPs and industry has accessibility to different options such as OLP software when conceiving the programming of a robot cell (refer to Section 2.1). Notwithstanding, the available path programming algorithms have not been fully explored on those software interfaces. A reason for that, and being one of the main drawbacks noted from the state of the art, is the lack of incorporation of the current knowledge representation such as CAD formats and the existing path generation algorithms into the OLP method. Figure 2.6 depicts the four knowledge areas in robotics required as state of the art for enhancing robot programmability for manufacturing purposes. The proposed synergy among these knowledge areas is the contribution of this thesis.

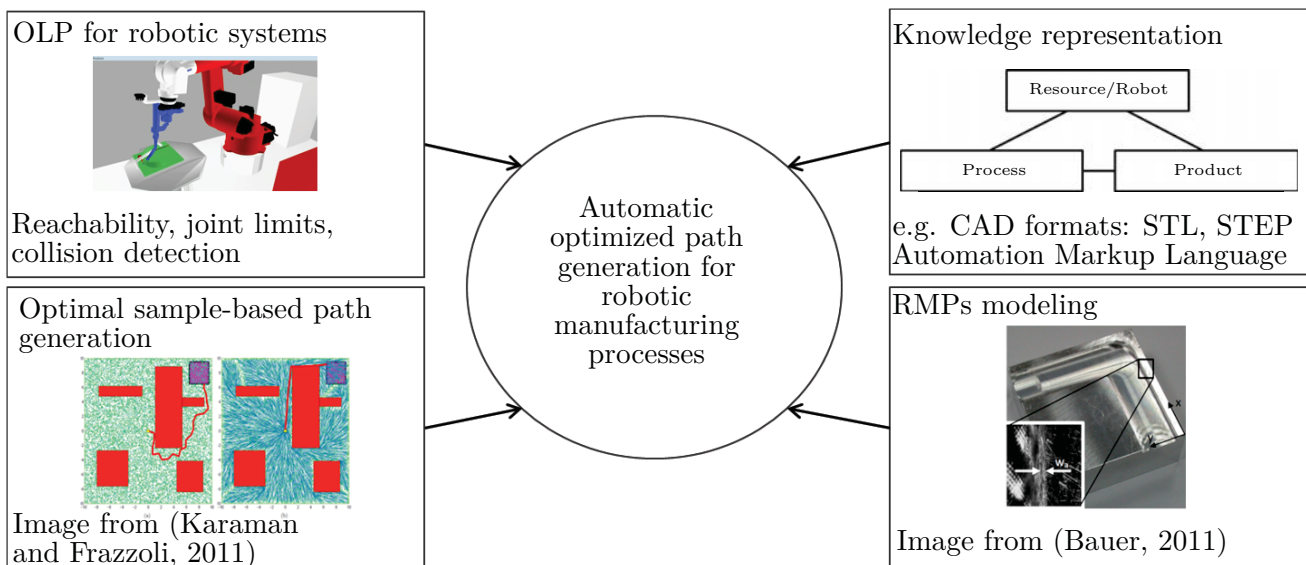


Figure 2.6: Synergy among knowledge areas of robotic manufacturing as potential for the automatic optimized path generation.

Confirmation of this deficiency is for instance observed from the description of using OLP provided by Perzylo, Griffiths et al. (2015, p.2008): “Industrial solutions such as Delmia or Process Simulate offer sophisticated robot programming interfaces that scale to a full production line with multiple robots. However, they require the programmer to specify robot tasks on a low level that requires expertise in robotics, thereby making these solutions difficult to use for untrained shop floor workers. The operators typically have to train for several weeks to learn how to use these interfaces. Even small changes in the robot program require significant programming effort.”

An important identified source of knowledge representation on this synergy is the CAD product models (refer to Subsection 2.2.2). The potential of using them into planning algorithms, such as the sample-based, is highlighted for reaching the goal of automatic optimized planning. Pires (2006, p.2) manifested also his interest in such a system: “It would be very interesting if the information from CAD files could be used to generate robot programs. That is, the CAD application could be the environment used for specifying the way the robots should execute the required operations on the specified parts.”

Besides, not just the geometrical knowledge representation of the product must be embedded into planning algorithms. Instead, also the notion of optimality with respect to the performance of the intended process should be approached for conceiving more complete systems. This lack of optimality has been pointed out also by other authors. Pan, Polden et al. (2012, p.91) state that “as most of the existing OLP software is not able to provide an optimal solution automatically. [...] Although some components for path planning and process optimization, such as collision detection, layout planning, time measurement, etc. are available, a complete path planning function does not exist and is fully relied on the process knowledge of the programmer.” Similarly, Nilsson, Johansson et al. (2005, p.2) said that “the generated motions from ideal CAD data have to be adjusted to current workpiece and process requirements. Thus, the adaptation of the robot requires adaptation and optimization of trajectories and process data.”

In regard to OLP, industry counts on highly developed commercial and open-source robot simulation software which allows modeling of robot cells, collision checking, and mapping of robotic dependent properties such as reachability, maximal joint limits, and singularities. However, as stated in COMET (2013, p.1), “what the CAM systems do lack is knowledge about the kinematics and dynamics of the robotic system; they are programmed to compute the tool paths for the simpler kinematics of standard machine tools. When confronted with a robot kinematic profile, the system cannot produce automatic collision-free path planning, automatically avoid singularities, and identify reachability. Also, the redundancy of certain robot axes is not used for potential optimization.” Therefore, characteristics of OLP systems must be further used in synergy with planning algorithms to solve this deficit.

Specifically for the sample-based planning, no approach or architecture relates the C-space with the robotic manufacturing process even though this need was already visualized by Verbarq (1998, p.2231) when he mentioned that “the relations between workspace and configuration space are still unsettled.” This deficiency is due to the main focus given by the community to other research and development topics such as the optimization of sequencing and assembly tasks which approach grid-based methods due to the difficulty of defining continuous spaces for these tasks, and the focus given to the planning in free-space for mobility, manipulating objects, and grasping.

To conclude, in a general approach, the specific process models must be linked with the path planning process in order to conglomerate the required knowledge for manufacturing with robots. The PPR model is identified as a well-defined and developed categorization for approaching this linkage. This was also pointed out by Kuss, Diaz Posada et al. (2016, p.350) when stating that “the models of specific manufacturing processes, required for a physical planning of robotic paths, are currently not integrated well with existing model formulations in manufacturing engineering. One reason is that research in this discipline is focused on the development of new algorithms and not on the formalization of knowledge for usage in automated robot program planning”.

2.7.2 Problematic and Motivation in the Studied Robotic Manufacturing Processes

The following subsection presents the problematic and motivation in the studied Robotic Manufacturing Processes (RMPs). The state of the art of the studied RMPs, i.e. welding, milling, and deburring were presented in Sections 2.4, 2.5, and 2.6 respectively. The specific use-cases are defined in Chapters 4, 5, and 6.

- *Optimized Collision Avoidance in Robotic Welding*

Robotic welding is one of the most performed tasks in manufacturing with Industrial Robots (IRs). According to the IFR Statistical Department (2017), the proportion of operational stock of welding and soldering robots is of 26.5 % by the end of 2016, only exceeded by handling applications which could be considered less complex due to the fewer constraints involved in this process. Despite this, also as stated by the IFR Statistical Department (2017), about 75 % of welding robots were shipped to major car production facilities worldwide depicting that just large scale companies can afford the implementation and programming of these systems.

The robotic welding process has been widely investigated as specified in Section 2.4. Several methods and algorithms have been used in order to enhance the programmability as described in Subsection 2.4.1. Moreover, the collision detection and avoidance is a topic widely researched due to its intrinsic nature when using robots (refer to Subsection 2.4.2) and the optimization of the process has been approached as well as detailed in Subsection 2.4.3.

Nonetheless, the developments in collision avoidance have been done in the programming of transitions and auxiliary motions in free-space. Most of the reported development making use of grid-based and sample-based algorithms are intended for solving the transition paths and the sequencing of the welding process. To the best knowledge of the author and based on the literature review, no concrete approach with sample-based algorithms has been implemented for optimizing the welding process taking into account infeasibilities. Furthermore, all described applications are reported as single use-cases and are not part of a modular and a changeable architecture allowing optimization of several processes. Due to the previously mentioned arguments and as further contribution of this work, the first use-case proposed in this dissertation is the implementation and investigation of an optimized collision avoidance using optimal sample-based algorithms for the welding processes.

- *Optimized Stiffness for Robotic Milling*

As described in the literature, robotic systems have become more common in the industry due to their low cost, flexibility, and versatility. However, the main challenge for this technology, in order to be also used extensively in machining operations, is its lack of absolute accuracy. One of the factors affecting the absolute positioning accuracy during machining is the lack of robot stiffness, which is the criterion studied and optimized in this use-case. Moreover, another extra challenge of robotic machining is the time-consuming, expert-dependent, and complex programming process. For this process, qualified personnel with experience in machining operations and robot programming is always required. In addition, the optimization of the machining processes nowadays is mainly achieved due to manual intervention on the generation of programs using the teach-in approach.

Several authors pointed out the need of optimization in the literature (refer to Subsection 2.5.3). For instance, Tyapin, Kaldestad et al. (2016, p.5511) state that “the stiffness of a manipulator depends on the configuration, hence the stiffness could be found in different configurations and interpolation techniques could be used to compute the stiffness in all other configurations.” And the conclusions from Pan and Zhang (2008, p.453), which determine that “the pattern of the process structure deformation is related to all of the following parameters: robot configuration, the location in the workspace, and the direction as well as the magnitude of the process force”, which confirms the relevance of optimizing robot configurations with respect to the robot stiffness.

Even though several research works have been conducted on error source identification, their compensation (Subsection 2.5.1), and robot joint compliance modeling (Subsection 2.5.2), few studies are to be found regarding optimization in this area (refer to Subsection 2.5.3). The optimization noticed in literature is case-specific and not performed with sample-based algorithms which justify the relevance to investigate and research in this direction.

Taking into account the identified drawback, the need to increase robot accuracy while machining, the potentials of optimizing stiffness based on the robot configuration, and the state of the art, confirms the need for improvement in the area of robotic machining. This use-case is proposed as contribution for addressing automatic optimization of robot stiffness in order to minimize deviations due to the robot compliance in a machining task using the functional DoF of the milling process, implemented into a general and changeable approach.

- *Optimized Sensing Strategy for Robotic Deburring*

Robots are accessible and universal machines which could enhance the productivity when deburring is required. However, robots are still not widely used in industry for this purpose mainly due to the lack of accuracy and complex programmability. In comparison with the robotic milling process, more sensor-based applications have been proposed for the deburring task. Examples of offline (Subsection 2.6.1) and online (Subsection 2.6.2) programming and compensation have been referenced. The facility of sensing properly single defined product features, such as edges, with nowadays available sensors is one of the reasons for the online trend.

Sensor-based systems are becoming more popular in the industry and the scientific community due to the accessibility of these components. 2D and 3D sensors and computational systems such

as Programmable Logic Controllers (PLCs) or computers are nowadays available for industrial implementation. However, they are not widely implemented in SMEs due to their complex programmability. OLP systems are tools which could be used for facilitating and enhancing the programming experience also when sensors are available. Other authors such as Dietz, Schneider et al. (2012) share also this opinion, by finding that an offline program generation system which incorporates sensor information, CAD models, and human input can effectively reduce the overall programming effort.

Following these ideas, a system which programs itself automatically by taking into account optimized sensing will be of advantage for industrial applications. While programming sensor-based or sensor-guided systems, difficulties such as not seeing the targeted feature or the proper fixing of the visibility area in respect to the workpiece is a time-consuming and a tedious work for programmers. To the best knowledge of the author, there is up to date no development of such optimization sensing strategy for the robotic deburring processes. This sensing strategy implemented in the general approach is the main contribution of this use-case. Therefore, in order to improve the programming experience, a system which optimizes the sensing criteria automatically for a 2D sensor-guided deburring system is proposed as the third use-case of this dissertation.

- *Automatic Close-to-Optimal Workpiece Positioning for Robotic Manufacturing.*

A specific aspect on the robotic programmability is the workpiece positioning. Even though this is a general problem in the programming of robotic manufacturing task, only few and case-specific solutions are to be found in literature as reported in Subsections 2.4.4 and 2.5.3 for the welding and milling use-cases, respectively.

For the workpiece positioning most of the in the literature presented developments are related to the time-sequencing optimization or task planning and not to the optimization of the actual motion during the execution of the manufacturing process in respect to manufacturing criteria. All the reviewed publications are stand-alone implementations for specific use-cases. A general architecture and approach has still not been clearly defined. This deficiency is attributed to the lack of a proper model-based approach for handling different processes and optimization criteria in order to automatically optimize RMPs.

To overcome these deficiencies and to improve the programming experience of RMPs, as further contribution, the automatic close-to-optimal workpiece positioning for robotic manufacturing is addressed as an extension on the presented use-cases. The general approach for the close-to-optimal workpiece positioning is presented in Section 3.5. Exploration of the changeability and optimization in two processes is addressed. The first one is the welding case in which the optimal orientation in respect to the product feature is optimized in the path planning phase and a minimum joint movement and gravity cost are also used as criteria while positioning the workpiece by using the DoFs of a rotatory table (refer to Subsection 4.5.4). The second use-case presents the positioning of the workpiece in which the robot stiffness is optimized during path planning in a machining table (refer to Subsection 5.5.4).

3 The Automatic Optimized Offline Programming Approach for Robotic Manufacturing

This chapter describes the approach and architecture of the path generation for solving the stated problem. This approach is named Automatic Optimized Offline Programming (AOLP) for robotic manufacturing processes due to its automation compared to the offline programming. Section 3.1 introduces the proposed model-based description of the robotic manufacturing process. Section 3.2 defines the robotic manufacturing process configuration space. Section 3.3 formulates formally the AOLP problem. These sections offer key definitions for detailing the approach and architecture. Section 3.4 describes the approach based on the definition of the model-based structure. Section 3.5 extends the general approach with the automatic close-to-optimal workpiece positioning. Section 3.6 outlines the general architecture of the path generation. Finally, Section 3.7 concludes this chapter.

3.1 Proposed Model-based Description of the Robotic Manufacturing Process

As identified in Section "Proposition" (2.7), the Product, Process, and Resource (PPR) model is a well-defined and developed categorization for describing Robotic Manufacturing Processes (RMPs). Therefore, this model is adopted and further developed in this contribution. Some of the existing models are adopted and some are further developed in order to establish the approach and the architecture. The following sections detail the product, the process, and the resource models. Moreover, the definition of feasible region is provided. Figure 3.1 depicts the general model-based description of the RMP which constitutes the input for the approach and architecture. Each input is introduced in the next subsections (from 3.1.1 to 3.1.3).

The RMP is defined with homogeneous transformations (T) describing the relative poses of the PPR components. The rotation matrix of the homogeneous transformation is denoted with the symbol R . The world coordinate system (w) is fixed. The pose of the product p is expressed in the world frame by the transformation T_w^p . From the product frame, the process frame ($proc$) is defined with the transformation T_p^{proc} constituting the transformation chain of the RMP. The robot TCP is commanded to the T_w^{proc} frame in order to perform the task. Figure 3.2 depicts the described transformations.

The approach and implemented architecture compute automatically the transformations taking into account the PPR model-based description. The transformation T_w^p is computed taking into account the CAD model of the product, the product features selected by the end-user, and the equations associated to each product feature. This transformation constraints the desired translation and orientations of the

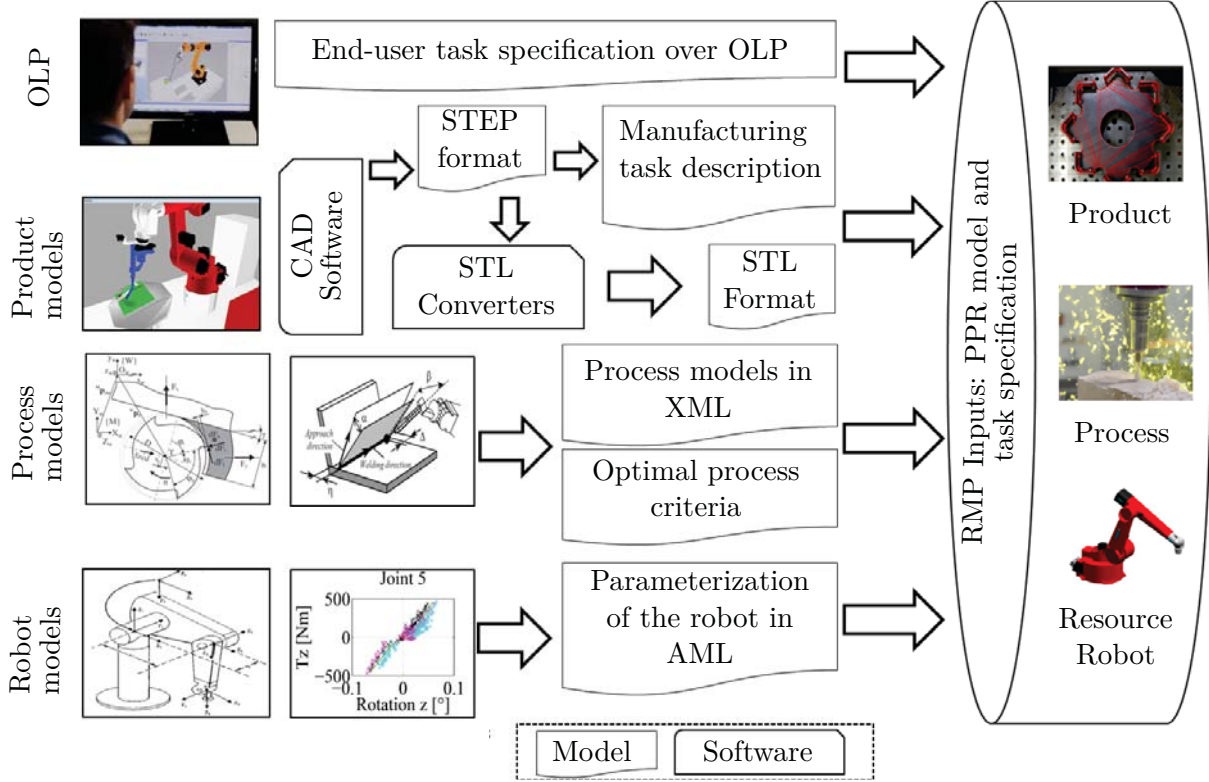


Figure 3.1: Approach and architecture inputs: model-based description of the RMP. With images from (Craig 1989, Dolgui and Pashkevich 2004, Slavkovic, Milutinovic et al. 2014, Schneider, Diaz Posada et al. 2015).

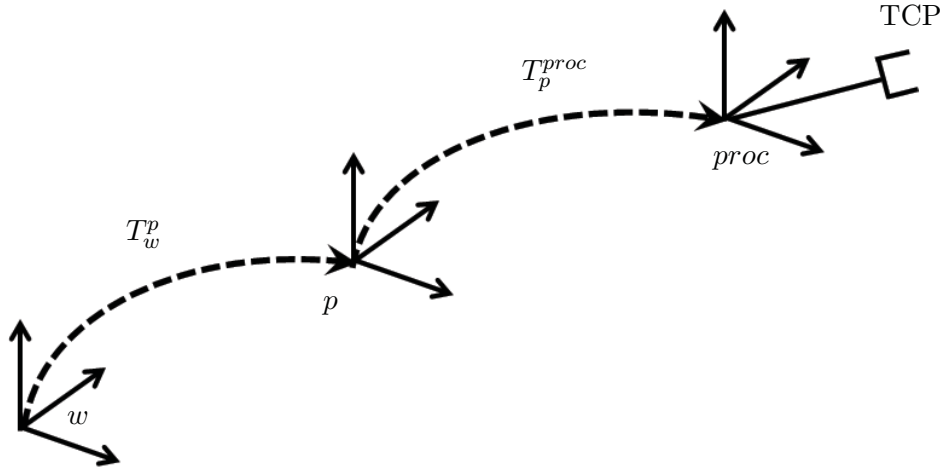


Figure 3.2: World, product, and process coordinate systems and their transformations.

product coordinate frame. Two interpretations of the transformation T_w^p are defined leading to different strategies (the single and the multiple problem strategy) for defining the path planning problem. Refer to next subsection for further detail of the strategies. The transformation T_p^{proc} is defined taking into account the process model specified in an XML format. This transformation specifies the constraints and DoFs of the process. Based on this model-based structure, detailed specified in next subsections for each of the PPR components, the automatic interpretation and planning takes place into a defined C-space for computing the RMP optimized paths.

3.1.1 Product Model

The product model is the representation of the workpieces which have to be manufactured. The product models used in this approach and architecture are based on the CAD formats of the workpieces. These models are denoted as (P). Parametric and mesh models (i.e. STEP and STL, respectively) are used in order to take advantage of each modeling type.

In this implementation, STL files are used for the final visualization in the simulation and for the collision detection as described in Subsection 3.2.1. The mesh models have as an advantage the characteristic of representing surfaces better by tiling them into other shapes such as triangles.

The STEP models are adopted for identifying geometric features such as points, lines, curves, and circumferences, among others. This is a key factor in the implementation of this approach and architecture. State-of-the-art software allow the modeling of products based on the specification of these features and their encoding in a semantic way in formats such as the STEP format.

In the implementation of the optimized path planner, STEP files are used for the generation of the manufacturing task description as presented by Kuss, Ksensow et al. (2017). The manufacturing task description offers, based on the STEP model of the product, the possible features which could be manufactured. This information is linked to the simulation environment for allowing the selection of product features from the simulation. This selection in conjunction with the process model setting (refer to Subsection 3.1.2) and the definition of the validity functions (refer to Subsection 3.2.1) defines the task specification.

A further contribution to the existing parametric models of products is the implementation of mathematical equations in software for their interpretation in a path planning problem. Each different feature in a product model is related to a function which generates an equation symbolically in software. These features could be for instance lines, curves, or circumferences describing the product. Parameterized equations are generated in the world coordinate system taking into account the transformation T_w^p and the information of the CAD file which is encoded relative to the p frame. Differentiation between the translational and rotational components of the homogeneous transformations is addressed in this model-based approach. The parameterization is explained below.

Parameterization of the Product Models

The variable k is introduced for parameterizing the equations describing the product. Parameterizations addressed in this work for the translation features are one-dimensional and can thus be represented by this variable. The constant k_g is designated to represent the end of the path over the variable k and the constant k_s for the start. The constant k_d is introduced for notating the end of the parametrization of a single product feature. In the use-cases presented, the constant k_d has been defined as 100. This constant is kept as defined over all the path plannings and shown C-spaces and costmaps in this document. The constant 100 is introduced to represent the parametrization as a percentage scale [%] over the product feature. This constant can be changed in the software if desired. For each feature, the length denoted as l can be computed from the CAD geometry and the assigned model-based equation in case of being required.

The simplest example of product features is for the linear ones. The description of the translational component of the homogeneous transformation for these features consists of the parameterized line equations in the Cartesian space by having as input the start (s) and goal (g) position vectors ($\vec{V}$) of each of the line features ($\vec{V}_{sm}$ and $\vec{V}_{gm}$ respectively defined in the product coordinate system p). Each selected feature regardless of its type (e.g. linear or circumferential) is identified with $m \in \mathbb{N}$. The number of features selected by the end-user in the simulation is denoted by $u \in \mathbb{N}$. The global position of the line is known in the simulation due to the fact that the transformation from the world coordinate system to the origin of the product coordinate system is determined (T_w^p). Figure 3.3a depicts the linear feature. Equation (3.1) describes the parameterization of a linear product feature (below the Figure 3.3a).

The optimal or reference rotational definition of the homogeneous transformation for each product feature is defined as a specific model for each process using quaternions (q) depending on the manufacturing characteristics. For instance, if a linear feature model identified with m demands constant orientation, this is defined by having as input the start quaternion q_{sm} , and then computing the optimal rotation matrix. This is denoted as follows, $R_{sm} = \mathbf{R}_q(q_{sm})$. The function for calculating a rotation matrix from a quaternion is denoted as $\mathbf{R}_q$. The goal quaternion is denoted as q_{gm} .

Quaternions are addressed for the generation of equations of the product on this approach because this rotational representation is singularity-free as opposed to other descriptions such as the Euler angles. This is an important characteristic when interpolating or parameterizing rotations. Nevertheless, the end-user parameterizes conventions of orientation in Euler descriptions which are more intuitive. The setting of values for α , β , and γ being the rotations around $\vec{x}$, $\vec{y}$, and $\vec{z}$ respectively are accessible to the end-user for manipulating the product in an AML configuration format. Note that if required, the rotational component is also parameterized with respect to k . An example of this is the optimal rotation interpolation definition between intersections among product features (detailed described in the coming Subsection 4.2.1) or the definition of optimal orientations for circumferential features.

Further examples of features are the semi-circumference and the circumference. Figures 3.3b and 3.3c depict these features. For parameterizing the translation of these features, rotational information is also required. A convention is defined in the simulation environment in which the $\vec{x}$ vector is parallel to the tangent of the circumference and the $\vec{z}$ vector parallel to the normal vector of the circle for defining the product. Robot directions, defined with the $\vec{x}$ vector for instance, can be adapted by the end-user as desired. The center of the circle $\vec{c}_m$ and its circle radius $|r_m|$ can also be determined by geometry. These inputs are used in order to define the translational constraints of the semi-circumference and circumference parameterized with k as described in Equations (3.2) and (3.3), respectively (below Figures 3.3b and 3.3c).

Taking advantage of the product translation description, the orientation is also parameterized with k for defining the product orientation convention through all the feature. By having the start orientation $R_{sm} = \mathbf{R}_q(q_{sm})$ of the semi-circumference or the circumference, the orientations are parameterized w.r.t. the k dimension. As example for the semi-circumference, at the end of one feature ($k = 100$), the final orientation describing the product q_{gm} rotates 180 Degrees (Deg) around $\vec{z}$. For the circumference, this rotation is of 360 Deg. Figures 3.3b and 3.3c depict the parameterization of orientation for the circumferential features. The described parameterizations are set as equations in the transformation T_w^p .

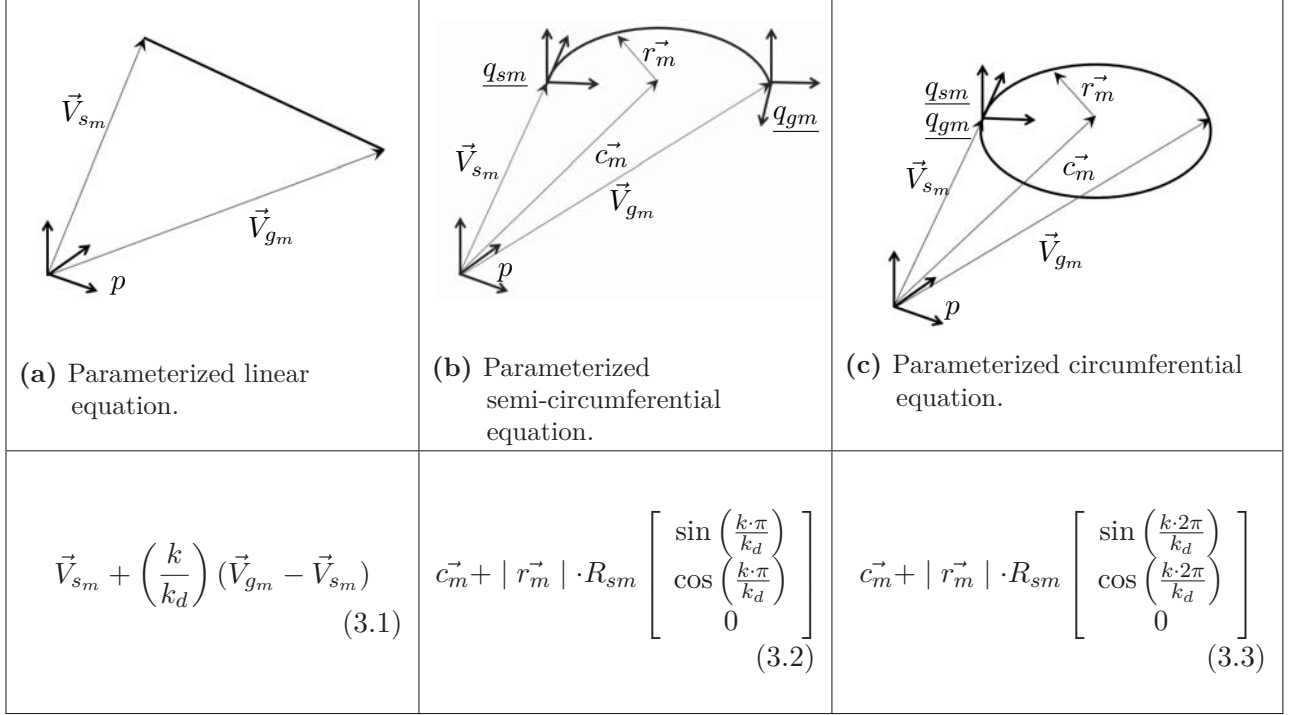


Figure 3.3: Product models of translational constraints for different features.

Product Interpretation Strategies for Multiple Continuous Features

For multiple continuous features, the path generation problem can be defined as a single problem by concatenating the selected features or as a multiple planning problem (one problem per feature). The approach has been programmed in order to analyze both cases by defining different strategies for interpreting the product (the multiple or single problem strategies). Figure 3.4 depicts these two product interpretations with the example of a product with three features (two linear and one semi-circumferential). The programmer defines if the interpretation must be approached as one or as multiple problems. The variable ϱ is introduced for this implementation, defining the number of problems. Product strategy interpretations are then addressed in the modular-based concept as the nature of the process demands it. This is one advantage of the proposed approach. The two different strategies are defined below:

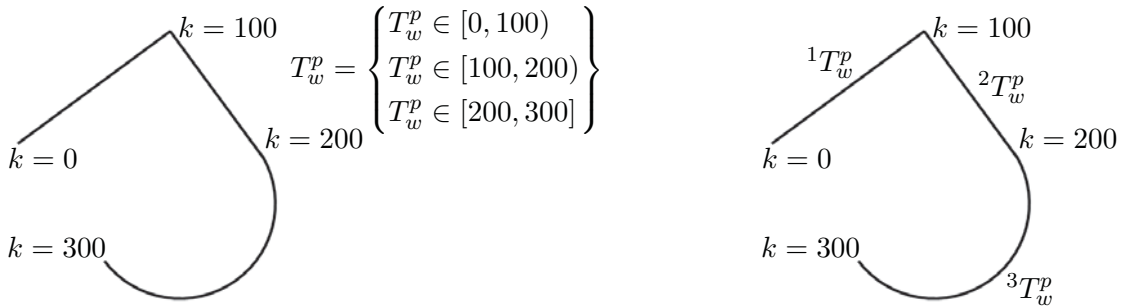


Figure 3.4: Illustrations of the different product interpretations.

3.1.1.1 Single product interpretation and problem strategy

For the single problem strategy, piecewise equations for the handling of multiple features on a product are defined. The piecewise equations concatenate the product models defining a single model for several features. The piecewise intervals are defined with respect to the specified k_d constant and the number of the features specified in the problem (u). Note that each product feature description in this research is a continuous model of the translation and the rotation defining the product.

Algorithm 1 describes the automatic interpretation of a single $u = 1$ or multiple-continuous linear features $u > 1$ with constant orientation interpreted using the single problem strategy. In both cases, the number of path problems is one ($\varrho = 1$). The output of this algorithm is the transformation world to product $T_w^p(k)$ parameterized in k with u number of piecewise equations. This transformation is denoted with Equation (3.4) Note that the output of this algorithm is an equation. The automatic generation of equations related to the products is one of the key features on the implementation of the model-based description in the software. The translation and rotational components vary depending on the type of model associated with each product feature. In Algorithm 1, parameterized linear components are specified as an example. However, if other feature types are selected, such as the semi-circumference, its specific equation is then defined in the translational component of T_w^p . If the rotations are also parameterized with respect to k , the equations are defined to the rotational component of T_w^p . Rotational parameterizations are defined for each use-case specifically as in the welding use-case (refer to Subsection 4.2.1).

Algorithm 1 Algorithm for automatic interpretation of linear features with constant orientation in a product with single problem strategy.

Input: $(\vec{V}_{s_1} \dots \vec{V}_{s_u}), (\vec{V}_{g_1} \dots \vec{V}_{g_u}), (\underline{q}_{s_1} \dots \underline{q}_{s_u}), u, k_d$

Output: $T_w^p(k)$ {automatic generated equation} (3.4)

```

1:  $T_w^p \left( DoF_p \right) = T_w^p(k) =$ 
2:  $\begin{bmatrix} \mathbf{R}_q(\underline{q}_{s_1}) & \vec{V}_{s_1} + (\frac{k}{k_d})(\vec{V}_{g_1} - \vec{V}_{s_1}) \\ 0 & 1 \end{bmatrix}, k \in [0, k_d]$ 
3: if  $u > 1$  then
4:   for  $m = 2$  to  $u$  do
5:      $\begin{bmatrix} \mathbf{R}_q(\underline{q}_{s_m}) & \vec{V}_{s_m} + (\frac{k}{k_d})(\vec{V}_{g_m} - \vec{V}_{s_m}) \\ 0 & 1 \end{bmatrix}, k \in ((m-1) \cdot k_d, m \cdot k_d]$ 
6:      $m = m + 1$ 
7:   end for
8: end if
9: return  $T_w^p(k)$ 
```

Advantage of using the single problem strategy is that just one planning problem is defined for treating several product features generalizing the implementation. Disadvantage of this strategy is the impossibility of implementing heuristics or algorithms to control the movements between features in the problem definition phase. Note that due to the piecewise equations and the definition of one path planning problem for treating several features, in case of discontinuities over the robotic manufacturing process Configuration Space (C-space) (detailed defined in Section 3.2) these are not properly handled

when changing from one feature to the next one. Therefore, this strategy is suitable when the C-space describes a continuous costmap such as the welding use-case (refer to Chapter 4). Note that these discontinuities are case-specific and appear on the C-space due to the relation between the definitions of the process manufacturing optimality and the product features.

3.1.1.2 Multiple product interpretation and problem strategy

If a multiple problem strategy is selected, then the product model is an array of homogeneous transformations $T_w^p(k)$ relating the model of each feature. The left superscript notates the identifier of the path generation problem as in Equation (3.5). If a multiple problem strategy is selected, then $\varrho = u$, defining for each selected feature a path generation problem.

A multiple problem strategy allows the proper avoidance of discontinuities of the costmap defined by the optimal definitions for the RMPs for some state cost and product definitions, such as in the milling use-case (refer to Chapter 5). the avoidance of discontinuities is achieved by implementing a heuristic algorithm for constraining the goal state of one feature and the start state of the next one in the path problem definition phase. Here the orientation in $\mathbb{SE}^3$ and the evaluation of manufacturing optimality are constrained to not deviate significantly. As disadvantage, this strategy demands the software to iterate the interpretation, the path problem configuration, and the sampling and optimization, u numbers of times. This represents longer computational times for the sampling and optimization. The difference between sampling and optimization computational times for both strategies can be observed in the robotic milling use-case (refer to Subsection 5.5.2).

$$\begin{pmatrix} {}^1T_w^p \\ {}^2T_w^p \\ \vdots \\ {}^{\varrho}T_w^p \end{pmatrix} \quad (3.5)$$

3.1.2 Process Model

The process model in this thesis is defined geometrically and through state and motion cost formulations as described in next subsections.

3.1.2.1 Geometrical Definition

The process model (PROC) is geometrically defined with the homogeneous transformation denoted as $(T_p^{proc}(\cdot))$, meaning the transformation from product to process. The orientation is defined using the Euler convention, which is more intuitive for the end-user for defining the constraints and DoFs of the manufacturing process. The description is performed locally, meaning with respect to the coordinate system of the features of the product.

A fixed transformation from the product to the process is defined with the constants $(x, y, z, \alpha, \beta, \gamma)_{proc}$. The DoFs of the process (DoF_{proc}) are defined depending on its nature and set as the allowed translations and rotations of the transformation $T_p^{proc}(\cdot)$, i.e. $(\delta x, \delta y, \delta z, \delta \alpha, \delta \beta, \delta \gamma)_{proc}$. Note that depending on the process some of these DoFs may be fixed. Equation (3.6) states the homogeneous transformation from product to process of a general manufacturing process. The function $\mathbf{R}_{\text{eul}}$ calculates a rotation

matrix from the angles describing the orientation given in Euler convention. The fixed transformation and the DoFs including their ranges are defined via XML format. The ranges are defined by specifying the maximal and minimal deviations for each DoF (e.g. α_{max} and α_{min} for the rotational DoF $\delta\alpha$). Figure 3.5 depicts the process model, the transformation product to process T_p^{proc} and one rotatory DoF of the process $\delta\alpha$, as example with a welding torch and a workpiece.

Examples of the XML process models are given in Listings 4.1 and 5.1 for the welding and milling process in Chapters 4 and 5, respectively. The XML file with the process parameters is linked from a general AML file which describes the robot cell.

$$T_p^{proc} \left(DoF_{proc} \right) = T_p^{proc} \begin{pmatrix} \delta x, \delta\alpha, \\ \delta y, \delta\beta, \\ \delta z, \delta\gamma \end{pmatrix}_{proc} = \begin{bmatrix} \mathbf{R}_{eul} \begin{pmatrix} \alpha + \delta\alpha \\ \beta + \delta\beta \\ \gamma + \delta\gamma \end{pmatrix}_{proc} & \begin{bmatrix} x + \delta x \\ y + \delta y \\ z + \delta z \end{bmatrix}_{proc} \\ 0 & 1 \end{bmatrix} \quad (3.6)$$

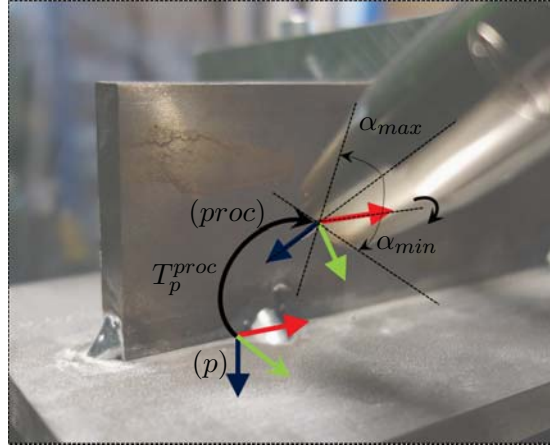


Figure 3.5: Illustration of the process model and the product to process transformation T_p^{proc} .

3.1.2.2 State and Motion Cost

The state and motion cost are defined in a general way in this subsection. These concepts are intrinsically related to the manufacturing process and describe the evaluation of optimality of a robotic manufacturing process. Specific definitions of state and motion cost for each manufacturing use-case are given in Sections 4.3, 5.3, and 6.3 for the robotic welding, milling, and deburring use-cases.

State cost

The state cost defines how expensive a specific robot configuration is for the manufacturing process. The state cost definition for the proposed approach defines the optimum at zero or at the minimum possible value. This definition is of importance for the reuse of algorithms in the approach. Other definitions could be applied, but this will represent changes in the algorithms. Figure 3.6 depicts two state costs s_i and s_{i+1} in a two dimensional C-space defined by $\delta\alpha$ and k . The dimensions $\delta\alpha$ and k have been defined as example. The extra dimension defines the state cost C_s of each state which is computed with the function $\mathbf{C}_s(\cdot)$. Note that the state cost can be defined for any dimensionality. The

state cost is a discrete evaluation of the optimality for just one specific state. The evaluation of the state cost through all the C-space defines the denominated RMP costmap or state elevation map for the manufacturing process model.

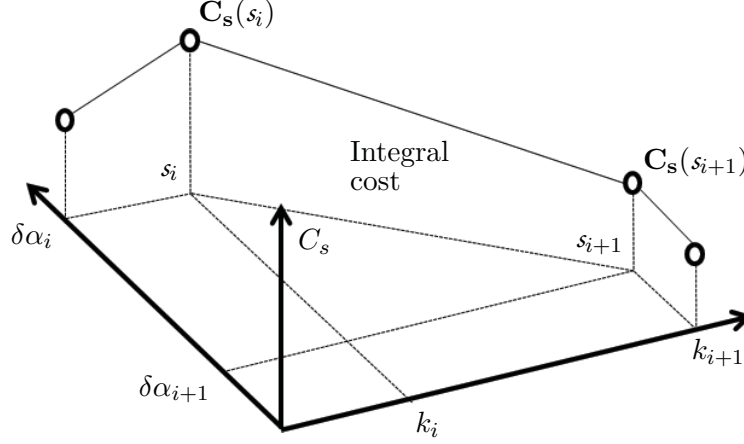


Figure 3.6: Illustration of the state cost.

Manufacturing Motion Cost

Based on the definition of state cost, the manufacturing motion cost C_M computed with the function $C_M(\cdot)$ is defined as the cost of the robot movement over the interconnected states through all the path (σ). This cost is specifically related to the manufacturing process itself.

Two main types of manufacturing motion cost are used in the presented use-cases, the integral and the mechanical motion cost. Further extension of these costs, available in the OMPL source code (Kavraki 2017), allow taking into account the k parameterization which defines the constraints of the product model over the path σ (refer to Section 3.1.1). The integral cost charges the motion based on the area defined under connected states (see the lines connecting the states forming areas in Figure 3.6). For the welding use-case (refer to Subsection 4.3.2) and the sensing strategy optimization (refer to Subsection 6.3.2) the integral motion cost has been implemented. The integral motion cost is denoted as follows,

$$C_M(\sigma) = \int C_s dk. \quad (3.7)$$

The mechanical cost charges the motion depending on the slope between states for keeping the path on the valleys of the RMP costmap. The mechanical work motion cost has been developed to compute motions that minimize the climbing of high-slope regions on elevation maps of the terrain (Jaillet, Cortés et al. 2008). As defined by Jaillet, Cortés et al. (2008, p.2146), the idea of the mechanical work is that “positive variations of the parametric cost function can be seen as forces acting against motion and thus producing mechanical work. We propose to use this loss of ‘energy’ induced by the mechanical work for measuring the quality of a path. In the case of negative variation of costs, the system does not lose any energy. Then, a small penalty proportional to the distance is added in order to favor shortest paths of equal mechanical energy”. The mechanical cost is defined for a path σ , with $\sigma+$ as portions of the path with positive slopes and with ϵ as a small penalty proportional to the distance adapted from the definition proposed by Jaillet, Cortés et al. (2008) as follows,

$$\mathbf{C}_M(\sigma) = \int_{\sigma+} \frac{\delta C_s}{\delta k} dk + \epsilon \int_{\sigma} dk. \quad (3.8)$$

The mechanical cost is implemented in the milling use-case where it is used in synergy with the T-RRT algorithm for optimizing the robot stiffness (refer to Subsection 5.3.2). Selection of the mechanical cost for this use-case is due to the need of avoiding high-slope regions over the RMP costmap. Note that for any use-case presented in this research both costs could be adopted.

Backward motion policy cost

Several manufacturing processes demand movements just in one direction including the three use-cases presented. Therefore, a policy cost C_p defined with the function $\mathbf{C}_p(\cdot)$ is introduced in order to avoid undesired backward motions. This policy cost is implemented taking advantage of the parameterization k , which constrains the translation of the product features. Thus, if a movement is attempted by the sample-based algorithm in the negative direction (from k_{i+1} to k_i as depicted in Figure 3.6), then an infinite cost is assigned as described in Algorithm 2 in order to avoid that motion. Note that the infinite cost is assigned in software with a large number.

Algorithm 2 Backward motion cost policy.

Input: k_i, k_{i+1}

Output: C_p {cost policy if backward motion must be avoided}

- 1: **if** ($k_{i+1} < k_i$) **then**
 - 2: **return** $C_p = \mathbf{C}_p(k_i, k_{i+1}) = \infty$
 - 3: **else**
 - 4: **return** $C_p = \mathbf{C}_p(k_i, k_{i+1}) = \emptyset$
 - 5: **end if**
-

Angular velocity motion cost

The angular velocity motion cost is defined as the rate of change between the k parameterization of two generated samples (k_i and k_{i+1}) for the angle describing a rotatory DoF of the process. Based on this definition, this motion cost is implemented in order to constrain the angular velocity with respect to (w.r.t.) the k parameterization. For setting this, a maximum admissible angular velocity ($|r_{max}|$) and the desired angular velocity ($|r_d|$) both w.r.t. the k parameterization, defined with units $[\frac{Deg}{\%}]$ for one rotational DoF, have to be specified by the end-user. Constants $|r_{max}|$ and $|r_d|$ are parameterized as positive quantities, representing positive and negative ratios in the path planning.

Given these inputs, equations are defined for penalizing undesired ratios. For implementing these equations in a general way, the given end-user inputs are normalized in a range from 0 to 1. The ratio border value is defined as $b_v = \frac{|r_d|}{|r_{max}|}$ representing the desired angular velocity in the normalized scale. The normalized angular ratio (a_r), for one rotational DoF ($\delta\alpha$ for instance), is defined as in Equation (3.9) and dictated by the interconnection trials done by the sample-based algorithm. Using the previous definitions, a ratio cost factor (r_c) is defined for constraining the ratios w.r.t. k , as defined in Algorithm 3, where r_1 and r_2 are constants and h the power factor.

Equations and parameters defining the ratio cost factor are heuristically defined based on the common manufacturing costmaps of the C-spaces for a certain process. Different equations and parameters can be defined depending on product, processes, and cost functions by the developers of software modules.

The equations of Algorithm 3 have been parameterized with $r_1 = 1$, $r_2 = 100$, and $h = 2$; and tested for the welding use-case (refer to Subsection 4.5.2). Ratio cost factors for different ratio border values are plotted for the spectrum of the normalized angular ratio in Figure 3.7. The normalization allows the generalization of this heuristic and therefore $|r_{max}|$ and $|r_d|$ are specified in the implemented use-cases (as in Subsection 4.5.2).

$$a_r = \frac{|k_{i+1} - k_i| \cdot |r_{max}|}{|\delta\alpha_{i+1} - \delta\alpha_i|} \quad (3.9)$$

Algorithm 3 Algorithm for computing the ratio cost factor.

Input: a_r, b_v, h

Output: r_c {ratio cost}

- 1: **if** $(a_r > 1) \vee (a_r < 0)$ **then**
 - 2: **return** $r_c = \infty$ {ratio cost policy}
 - 3: **else if** $a_r \leq b_v$ **then**
 - 4: **return** $r_c = \frac{-r_1 \cdot (a_r - b_v)^h}{(b_v)^h} + 2$ {ratio cost for rates slower than b_v }
 - 5: **else if** $a_r > b_v$ **then**
 - 6: **return** $r_c = \frac{r_2 \cdot (a_r - b_v)^h}{(1 - b_v)^h} + 2$ {ratio cost for rates faster than b_v }
 - 7: **end if**
-

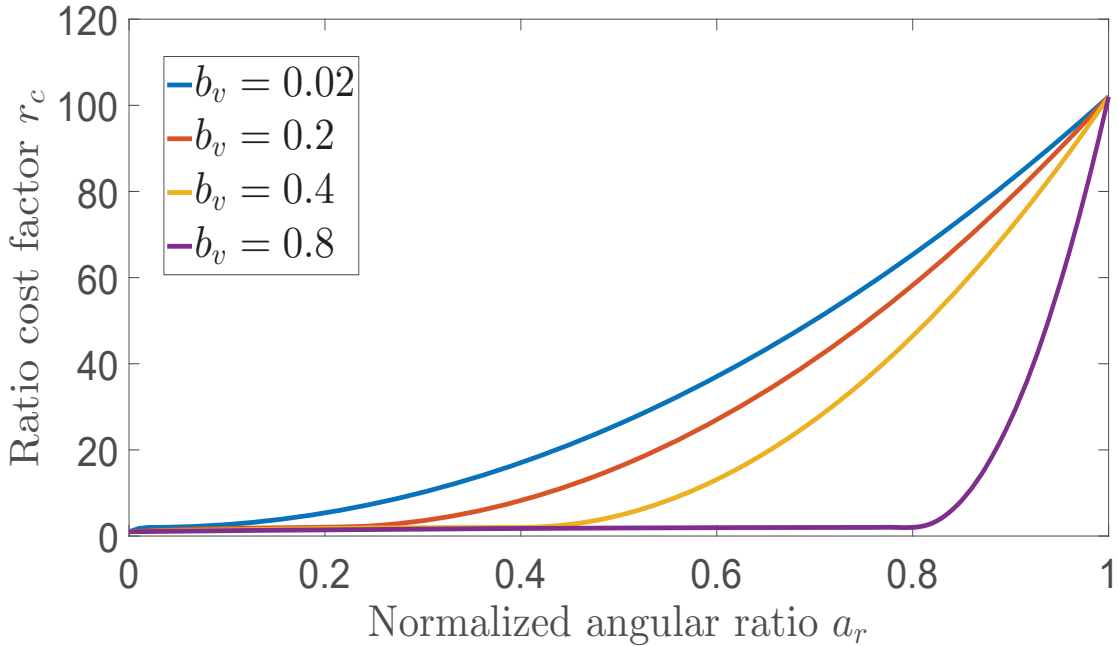


Figure 3.7: Ratio cost factor r_c versus the normalized angular ratio a_r for different ratio border values b_v in order to constrain the angular velocity w.r.t. k .

Total Motion Cost

The total motion cost is the weighted sum of the manufacturing motion cost, policies, and further costs implemented in order to constrain the motions of the planning approach. The manufacturing motion cost and the backward motion policy cost (C_p) were described above. Moreover, an angular velocity motion cost ($r_c \cdot C_M$) has been also implemented to constrain angular velocities taking into account the costmap of the RMP C-space. Detailed definitions of these costs were given above.

Weighting factors are provided in order to define the prioritization of the optimization. The weighting factor w_M , set between 0 and 1, defines the average in which the manufacturing motion cost must be taken into account with respect to the angular velocity motion cost. The weighting factor w_p , set as 0 or 1, defines if backward motions must be avoided. The total motion cost C_t and its function $\mathbf{C}_t(\cdot)$ is defined as the sum of the motion costs and policy costs prioritized with weighting factors as defined by the end-user as denoted in the following equation,

$$C_t = \underbrace{w_M \cdot C_M}_{\substack{\text{Weighted manufacturing} \\ \text{motion cost}}} + \underbrace{(1 - w_M) \cdot r_c \cdot C_M}_{\substack{\text{Weighted angular} \\ \text{velocity motion cost}}} + \underbrace{w_p \cdot C_p}_{\substack{\text{Backward motion} \\ \text{policy cost}}} \quad (3.10)$$

The definition of state and motion costs and the transformation T_p^{proc} in which the constraints and DoFs of the process are specified allow the description of the manufacturing process. These definitions are implemented in the modular-based structure in order to allow the changeability of process models. Specific process transformations and state and motion costs are defined in Chapters 4, 5, and 6 for the robotic welding, milling, and deburring use-cases.

3.1.3 Robot Model

The robot models (ROBOT) used in this work are adopted from existing models in the literature as stated in the state of the art (refer to Subsection 2.2.4). The robot Denavit-Hartenberg (DH) parameters for describing the specific robot kinematics are defined based on the information obtained from the robot provider datasheet. Rotations of joint servomotors are described around the $\bar{z}$ axis of each joint frame. This setting is performed using the AML file which specifies the robot cell. The spanned configuration space due to the robot joints is defined as $\mathbb{J}$ and named the robot joint space. Another robot model used is the robot joint compliance model which is described in detail in Subsection 5.2.4.

The forward kinematics is defined as $\underline{x} = \mathbf{FK}(\underline{j})$ in which $\underline{x}^T = [x, y, z, \alpha, \beta, \gamma]_{robot}$ defines a pose of the TCP in $\mathbb{SE}^3$ with x , y , and z for the translation and α , β , and γ for the rotation around the translation components, respectively. The notation $\underline{j}_{[6x1]}$ represents the robot joint positions in joint space (being of 6 dimensions for an industrial manipulator). Each joint frame is denoted with j following the number referring to the joint. The forward kinematics is parameterized and implemented using the Denavit-Hartenberg (DH) notation (Craig 1989, Siciliano, Sciavicco et al. 2010).

The inverse kinematics is defined as $\underline{j} = \mathbf{IK}(\underline{x})$ and it has been implemented with the geometrical approach similar to the one described in (Siciliano, Sciavicco et al. 2010) decomposing the problem in

two decoupled problems. Specifically, the solution for manipulators with a spherical wrist has been implemented in the simulation environment. Due to the fact, that the last three consecutive revolute joint axes intersect at a common point, first the robot wrist position is calculated from the TCP pose. Then, the inverse kinematics for the first three joints is computed by taking into account the solution of the 3 DoF anthropomorphic arm. Finally, the inverse kinematics of the orientation (three last joints) is computed as the solution of the spherical wrist.

Due that the inverse kinematics has multiple possible solutions, its software function is constrained to one solution by setting the front/back position, the elbow up/down or the wrist up/down configurations. Results presented in this work are configured with a front position, elbow up, and wrist down configurations, which resembles manufacturing operations in the evaluated cells. Moreover, all automatically generated programs for manufacturing have been run in Cartesian space because the real calibrated parameters of the robot kinematics in their controllers are unknown and the final manufacturing process quality is highly dependent on this calibration. In the proposed path planning approach robot singularities could be avoided by charging the movement between two robot configurations when the robot joint velocities overpass a higher threshold.

3.2 Robotic Manufacturing Process Configuration Space ($\mathbb{RMP}^n$)

The Robotic Manufacturing Process (RMP) Configuration Space (C-space) is defined as a C-space in which the product translational constraints and the optimal orientations through a product feature are set in one dimension and the further DoFs of the process as further dimensions. Definitions of cost functions, which may include robot models, are associated to the C-space for evaluating each robot configuration in order to allow the optimization of RMP paths.

This novel configuration space is proposed for allowing and simplifying the computation of optimized paths for robotic manufacturing. This C-space is denominated the robotic manufacturing process configuration space ($\mathbb{RMP}^n$) and it is used jointly with the Cartesian space ($\mathbb{SE}^3$) and the joint space ($\mathbb{J}$). The dimensionality of this space is denoted with n . In the use-cases implemented, the maximal dimension of this space is four, one dimension constraining the product features and three rotational DoFs. Nevertheless, this space can be further extended to any dimensionality as required.

The functionality of this C-space is to relate motions with the product, process, and robot components of a manufacturing system. The main goal of the inclusion of this C-space is the simplification of the description of a manufacturing process. This is achieved by incorporating the PPR constraints into this C-space and setting the common DoFs of the PPR components as dimensions. Figure 3.8 depicts the involved C-spaces in the approach and their transformations.

The inverse kinematics $\underline{j} = \mathbf{IK}(\underline{x})$ is used for transforming poses $\underline{x}$ from $\mathbb{SE}^3$ to joint positions $\underline{j}_{[6x1]}$ in $\mathbb{J}$, and the forward kinematics $\underline{x} = \mathbf{FK}(\underline{j})$ for the other way around (both defined in Subsection 3.1.3). Similarly, the function for transforming from $\mathbb{RMP}^n$ to $\mathbb{SE}^3$ is introduced and denoted as $\mathbf{f}(\underline{s}) : \mathbb{RMP}^n \rightarrow \mathbb{SE}^3$, where $\underline{s}$ denotes a robot state. For the other way around, i.e. from the Cartesian space to the $\mathbb{RMP}^n$, no transformation is required on the implementations experimented in this thesis because the path planning process happens in $\mathbb{RMP}^n$. Instead, the information regarding the feasibility of a state (refer to Subsection 3.2.1) and the state and motion costs (refer to Subsection 3.1.2.2) are used in the sampling of this C-space for allowing the planning and the optimization.

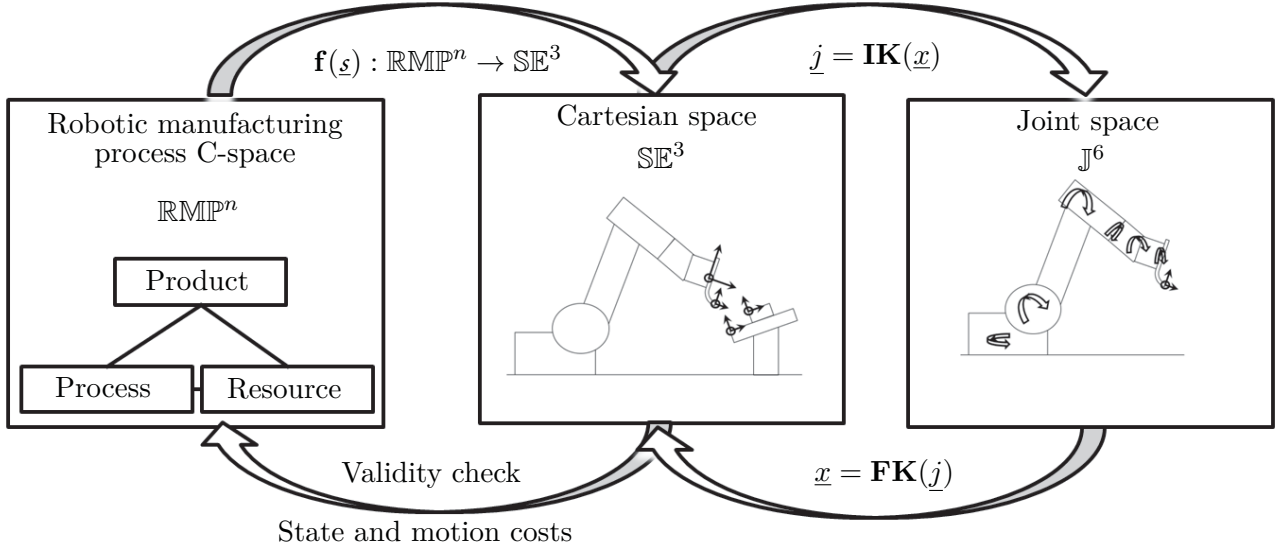


Figure 3.8: Configuration spaces involved in the optimal path generation.

The transformation from the RMP C-space to the Cartesian space ($f(\underline{s}) : RMP^n \rightarrow SE^3$) is defined based on the kinematic chain of the robotic manufacturing process. The product and process models are taken into account depending the intended manufacturing process. The DoF of the product (DoF_p), characterizing the constraints and desired orientations, is parameterized with k . This defines the transformation T_w^p as described in Section 3.1.1. The transformation T_p^{proc} is defined with the process constraints and its DoFs (DoF_{proc}). Equations for the translational components in the Cartesian space and the quaternions describing the rotations are automatically generated by the approach as denoted in the following equation,

$$f(\underline{s}) = T_w^p(DoF_p) \cdot T_p^{proc}(DoF_{proc}) = \begin{cases} x = \mathbf{f}_x(k) \cdot \mathbf{f}_x(DoF_{proc}) \\ y = \mathbf{f}_y(k) \cdot \mathbf{f}_y(DoF_{proc}) \\ z = \mathbf{f}_z(k) \cdot \mathbf{f}_z(DoF_{proc}) \\ q_1 = \mathbf{f}_{q1}(k) \cdot \mathbf{f}_{q1}(DoF_{proc}) \\ \vdots \\ q_4 = \mathbf{f}_{q4}(k) \cdot \mathbf{f}_{q4}(DoF_{proc}) \end{cases} \quad (3.11)$$

As stated in literature, the configuration space is a tool for formulating motion planning problems precisely (Latombe 1991). The C-space is defined by Lavelle (2011, p.13) as, “the set of all rigid body transformations that can be applied to the robot”. Up to date, no defined definition and/or approach exist for defining C-spaces for robotic manufacturing processes. This is one of the main contributions of this thesis. The task space is defined by Lynch and Park (2017, p.32) as “a space in which the robot’s task can be naturally expressed. For example, if the robot’s task is to plot with a pen on a piece of paper, the task space would be $\mathbb{R}^2$ ”. Similarly, the task space for robotic manufacturing processes can be defined as $\mathbb{R}^2$ or $\mathbb{R}^3$. The RMP^n differs from the task space defined by other authors (Lynch and Park 2017, Canudas de Wit, Bastin et al. 1996) because it relates product features entities (such as lines or semi-circumferences) into one dimension, process related DoFs into further dimensions,

and moreover includes the state cost for creating costmaps which represent the RMP for allowing planning and optimization using sample-based algorithms. Note that the product features entities are constraints interpreted as a DoF in the RMP C-space.

The main goal of the inclusion of the RMP C-space is the simplification of the manufacturing process. This simplification is achieved by methodologically and automatically transforming the translation constraints of the product and the optimal orientations with respect to the process into a DoF (the dimension defined with k) and by interrelating this DoF with the DoFs and constraints of the process.

As example, Figure 3.9 depicts an $\mathbb{RMP}^2$ and the mapping of samples to $\mathbb{SE}^3$. The product consists of two features: a semi-circumferential and a linear. The translational constraints and the optimal or reference orientations of the product are set in the k dimension in the $\mathbb{RMP}^2$. This example depicts the setting of the C-space with the single problem strategy. For the multiple problem strategy, an $\mathbb{RMP}^2$ for each feature would be defined. The k dimension is set up to 200 because two features are interpreted (100 percentage per feature). As example, the dimension $\delta\alpha$ is defined for the DoF of the process. The maximal and minimum deviations from the optimal or reference orientation are defined with the variable α_{max} and α_{min} , respectively. The following subsection describes the feasible region of the $\mathbb{RMP}^n$.

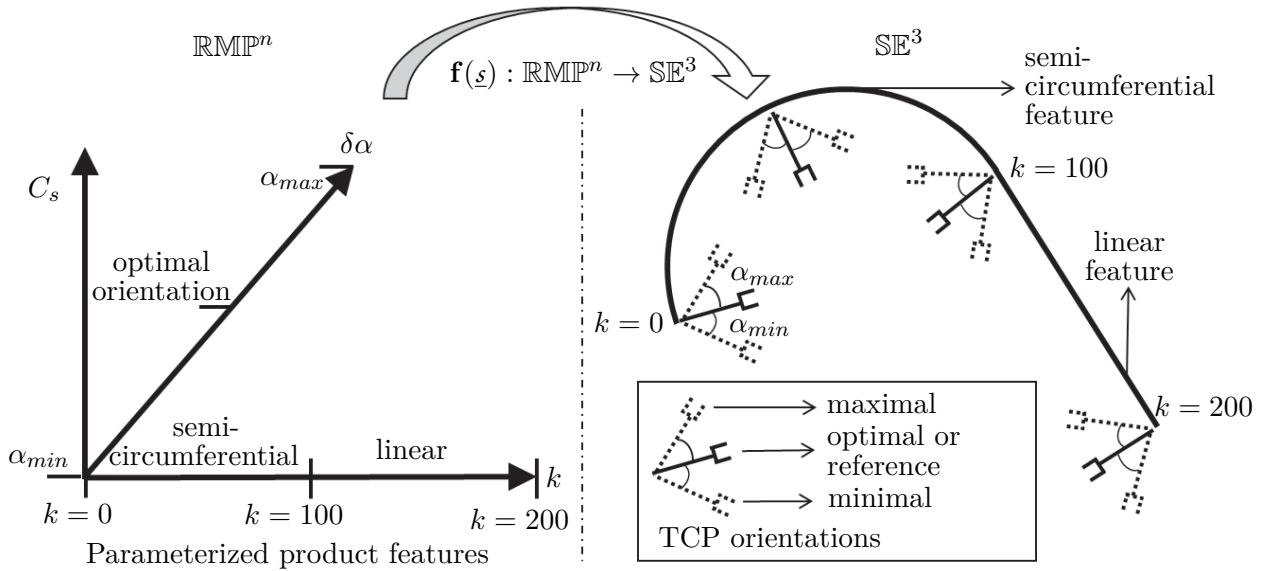


Figure 3.9: Example of the RMP C-space and the mapping of samples to the Cartesian space.

3.2.1 Feasible Region of the Robotic Manufacturing Process Configuration Space

The feasible region of the robotic manufacturing process configuration space is defined as all the states in the $\mathbb{RMP}^n$ that are mapped to valid robot configurations in $\mathbb{SE}^3$ through $f(\underline{s})$. The feasibility is defined as the binary discrete decision (true or false) of the validity of a robot configuration. In this implementation, the functions of collision detection, maximal joint limits, and reachability are considered for defining the feasibility. Use of these functions is decided by the end-user on the software GUI (refer to Section 3.6). If more than one function is selected, a conjunctive decision is conducted.

The software function for determining feasibility is defined as $\text{IsValid}(\cdot) \rightarrow \text{True} \vee \text{False}$, stating this the validity or non-validity of a state. The infeasible region denoted $\mathcal{S}_{unfree}$ is the set of all non-feasible

states. The feasible region or workspace of the C-space is defined as $\mathcal{S}_{free} = \mathbb{R}^{MP^n} \setminus \mathcal{S}_{unfree}$. The validity functions which define this region are defined below.

Collision Detection

The implemented collision detection function is based on the "faster triangle-triangle intersection tests" (Devillers and Guigue 2002). This method computes whether or not two triangles in the Cartesian space intersect. An open-source code (Guigue and Devillers 2003) has been used in the simulation environment for implementing this method. The STL product models and the STL models of the robot joints and further objects in the scene, defined by triangles, are loaded into the simulation environment and their position and orientation can be manipulated as desired. In this way, collisions between entities such as the product and the robot can be computed.

First testing collisions within the bounding boxes allows to speed up the computation. The bounding boxes method determines a closed volume that completely contains the object. Heuristics functions are also applied to improve the computation, such as the one that robot joints can not collide if the joint limits are set. The validity function first identifies if the bounding boxes triangles collide. Just in case they do, the collision analysis between STL objects is also performed. The final result of this function is the true or false determination of collision for a given configuration of the objects in the workspace.

Maximum Joint Limits

The maximum joint limits function determines if the joints are in the defined range. The minimum ($^{min}j_i$) and maximum ($^{max}j_i$) joint limits are specified for each robot joint (i) based on the information provided by the providers. These limits are specified by the end-user in the AML file. This functionality avoids the generation of infeasible paths and programs in the OLP software saving programming time in the implementation phase. The following equation describes this function,

$$\text{true if, } \left\{ \begin{matrix} ^{min}j_i \leq j_i \leq ^{max}j_i \end{matrix} \right\}. \quad (3.12)$$

Reachability

The robot reachability is defined as the robot ability to move its joints in order to reach an specific pose. Figure 3.10 depicts the robot lengths and transformations involved in the computation of the reachability. The orientation of the robot base (rb) coordinate system is the same as the one of the world frame (w) depicted in black.

The robot reachability evaluation is defined here based on two heuristic evaluations. For this evaluation the $\vec{X}_w$ vector is defined from the coordinate frame of joint two to the coordinate frame of joint five of the robot. Note that $\vec{X}_w$ defines the start of the robot wrist. The first heuristic is if the x and y components of $\vec{X}_w$ vector are in between a range defined by ϵ_x and ϵ_y respectively $(|\vec{X}_{w,x}| > \epsilon_x) \wedge (|\vec{X}_{w,y}| > \epsilon_y)$. This range delimits an unreachable volume close to the robot base taking into account also the z dimension. The second heuristic is if the norm of the vector $\vec{X}_w$ is less than the addition of the lengths of joint two (l_2), three (l_3), and four (l_4) obtained for each robot from their data-sheets and configured for each use-case via the AML file $(\|\vec{X}_w\| < (l_2 + l_3 + l_4))$.

The vector $\vec{X}_w$ is computed taking into account the desired transformation from the robot base (rb) to the TCP (T_{rb}^{tcp}) which is composed of the desired rotation matrix R_{des} and position vector $\vec{X}_{des}$. The notation $\vec{V}_{j2 \rightarrow rb}$ defines the vector from the coordinate frame of joint two to the robot basis. The computation of reachability is summarized in Algorithm 4. DH notates the Denavit-Hartenberg data structure containing the robot lengths.

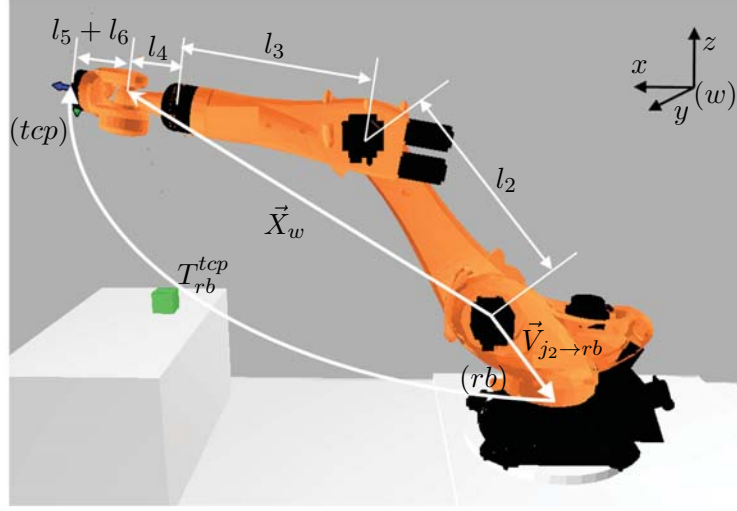


Figure 3.10: Robot lengths and transformations involved in the heuristic for the reachability computation.

Algorithm 4 Heuristic for reachability computation.

Input: $\left(DH, T_{rb}^{tcp} = \begin{bmatrix} R_{des} & \vec{X}_{des} \\ 0 & 1 \end{bmatrix} \right)$

Output: Reachability

$$\vec{X}_t = \vec{X}_{des} - R_{des} \cdot [0, 0, l_5 + l_6]^T$$

$$\vec{X}_w = \vec{V}_{j2 \rightarrow rb} + \vec{X}_t$$

$$C_1 = \text{True} \vee \text{False} \leftarrow \left((|\vec{X}_{w,x}| > \epsilon_x) \wedge (|\vec{X}_{w,y}| > \epsilon_y) \right)$$

$$C_2 = \text{True} \vee \text{False} \leftarrow \|\vec{X}_w\| < (l_2 + l_3 + l_4)$$

if $C_1 \wedge C_2 == \text{True}$ **then**

return Reachability = True

else

return Reachability = False

end if

3.3 Formal Problem Formulation

The following path generation problem is defined for addressing improvements in the programmability and reconfigurability of robotic manufacturing systems while providing manufacturing quality.

Given:

1. a workspace $\mathcal{W}$; for the object of this study $\mathcal{W} \in \mathbb{SE}^3$, where $\mathbb{SE}^3$ denotes the Cartesian space in which the robot cell is simulated.
2. a PPR model (refer to Section 3.1) consisting of
 - a) a Product model, denoted as (P);
 - b) a Process model, denoted as (PROC);
 - c) and a Resource model, which for the object of this study is a six joint articulated robot, denoted as (ROBOT);
3. a task specification (refer to Subsection 3.1.1);
4. and a total manufacturing cost $\mathbf{C}_t = \sum_{i=0}^{i=d} w_i \mathbf{C}_i(\cdot)$, where d is the number of different motion cost functions $\mathbf{C}_i(\cdot)$ and w_i their weighting factors (refer to Subsection 3.1.2.2).

Compute automatically and offline a feasible path: $\sigma : [k_s, k_g] \in \mathbb{RMP}^n$, with k_s , and k_g being the real numbers parameterizing the start and goal of a geometric product feature respectively, such that $\mathbf{f}(\sigma(s_s)) : \mathbb{RMP}^n \rightarrow \mathbb{SE}^3 = \underline{x}_s$ and $\mathbf{f}(\sigma(s_g)) : \mathbb{RMP}^n \rightarrow \mathbb{SE}^3 = \underline{x}_g$.

And optimize the path (σ^*) with respect to the total manufacturing motion cost for improving the RMP, $\mathbf{C}_t(\sigma^*) = \min\{\mathbf{C}_t(\sigma) : \sigma \in \mathcal{S}_{free}\}$.

Where s_s is the start state, constrained by $k = k_s$; and s_g the goal state, constrained by $k = k_g$. The notations $\underline{x}_s, \underline{x}_g$ are the start and goal optimized configurations in $\mathbb{SE}^3$ satisfying the RMP constraints; and the $\mathbb{RMP}^n$, the C-space interpreted based on the PPR model-based components.

After the formal definition of the problem, the proposed approach and architecture for solving it are described in the next sections.

3.4 Proposed Approach: The Automatic Optimized Offline Programming

The Automatic Optimized Offline Programming (AOLP) is an approach based on the further development of the widespread PPR model as described in Section 3.1 for configuring automatically state-of-the-art sample-based path generation algorithms in order to optimally solve a robotic manufacturing task. The approach is proposed with the goals of simplifying the programming of a robotic manufacturing process, and of decreasing the dependency of robot experts in the re-configuration phase. For describing the approach, the definitions of the PPR model (refer to Section 3.1), the $\mathbb{RMP}^n$ and its feasible region (refer to Section 3.2), and the formal problem formulation (refer to Section 3.3) have been introduced for facilitating its explanation.

The solution approach consist of three main steps as depicted in Figure 3.11: (1) the systematic interpretation of the manufacturing process using as inputs the model-based description of the PPR (from step 1 to 4 in Figure 3.12), (2) the automatic configuration of a state-of-the-art optimal sample-based algorithm and a path planning problem in the interpreted $\mathbb{RMP}^n$ (from step 5 to 8 in Figure 3.12)

and (3) the sampling and solving in the $\mathbb{RMP}^n$, for finally mapping the optimized path to the Cartesian or joint spaces and parsing it finally into a functional and optimized robot program (from step 9 to 12 in Figure 3.12). Figure 3.11 depicts also the model dependencies for each of the three main steps.

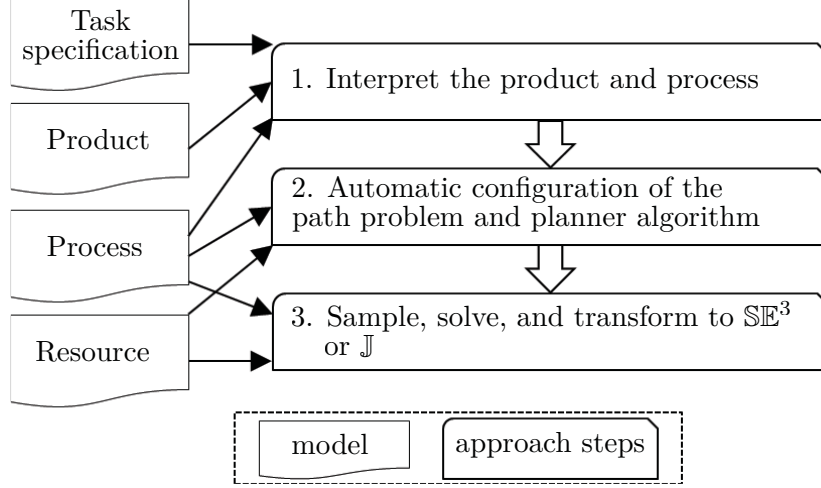


Figure 3.11: Inputs required by the general steps of the approach.

This contribution proposes to solve the problem formulated in Section 3.3 with the employment of optimal sample-based path generation algorithms, with the definition of optimization criteria in the C-spaces $\mathbb{SE}^3$ and $\mathbb{J}$ by evaluating the optimality in these C-spaces and giving back this information to the $\mathbb{RMP}^n$ for defining the costmap.

The solution approach for solving the defined problem then becomes the interpretation (notated as the software function $\text{Interpret}(\cdot)$) of the PPR classification (composed by the P, PROC, and ROBOT models) in the workspace ($\mathcal{W}$) for automatically setting and configuring an optimal sample-based algorithm (OSBA) for offline finding a feasible and optimized path σ^* with a final total cost C_t as denoted in Equation (3.13).

In this equation, the $\mathbb{RMP}^n$ denotes the robotic manufacturing process C-space of dimension n (refer to Section 3.2), $\underline{B}$ the bounds of the dimensions derived from the PPR model, s_s and s_g the initial and goal states respectively, the function $\mathbf{f}(\underline{s}) : \mathbb{RMP}^n \rightarrow \mathbb{SE}^3$ as defined in Section 3.2, $\mathbf{C}_t(\cdot)$ the total motion cost function, and the $\text{IsValid}(\cdot)$ function as defined in Subsection 3.2.1.

$$\begin{aligned} \text{Interpret}(\mathcal{W}, \text{P}, \text{PROC}, \text{ROBOT}) &\rightarrow (\mathbb{RMP}^n, \mathbf{f}, n, \underline{B}, s_s, s_g) \rightarrow \\ \text{OSBA}(\mathbb{RMP}^n(\mathbf{f}, n, \underline{B}), s_s, s_g, \mathbf{C}_t, \text{IsValid}) &\rightarrow (\sigma^*, C_t). \end{aligned} \quad (3.13)$$

The detailed required steps for the interpretation and for the automatic configuration of the sample-based algorithm are enumerated and described below. The workflow of the approach is depicted in Figure 3.12.

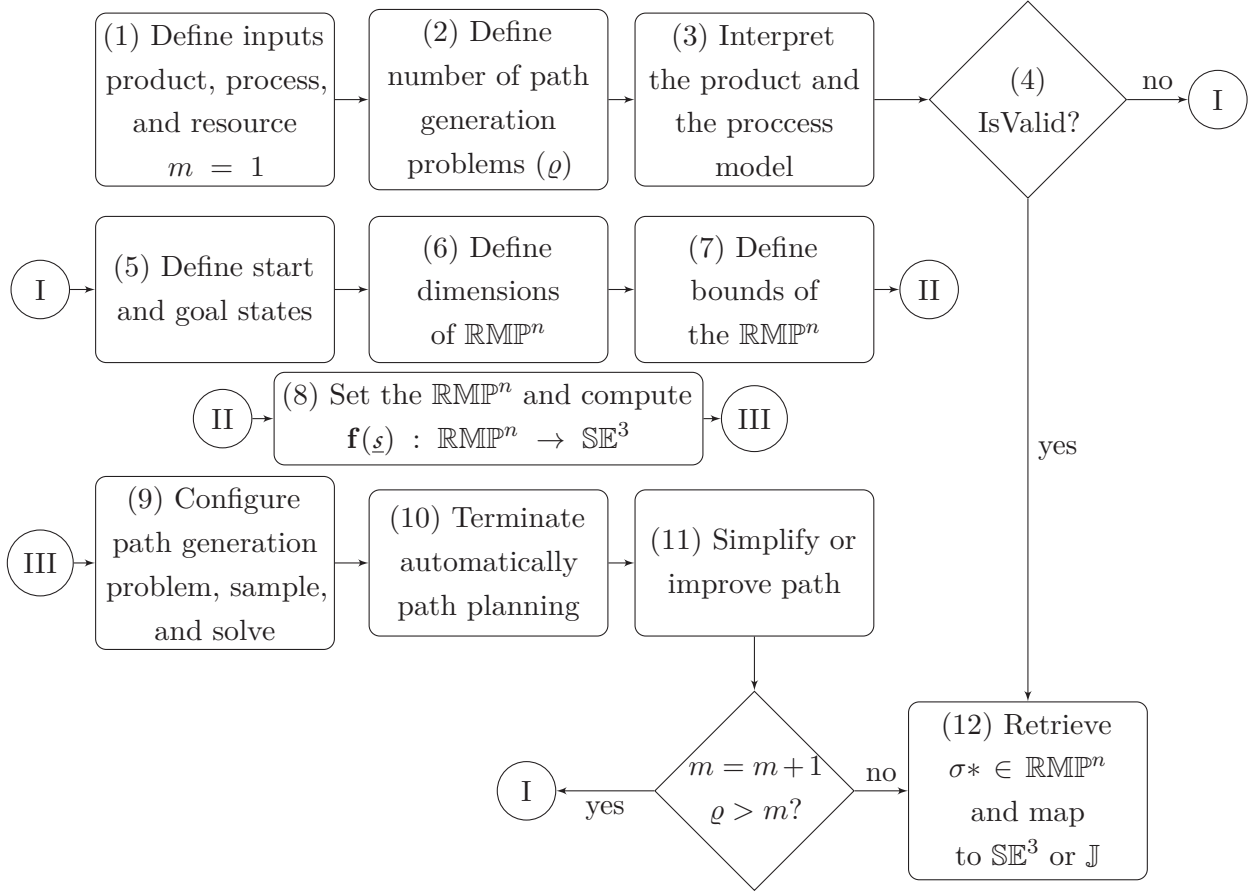


Figure 3.12: Workflow of the approach.

1. Define inputs:

The first step is the specification by the end-user of the PPR model and the task. The product is defined by loading the CAD file of the workpiece and by selecting the features to be processed in the simulation environment for determining the task specification. Moreover, the parametrization of the process model, the robot type and kinematic parameters, the selection of the optimization functions, and the definition of validity functions to take into account must be specified via XML, AML, and/or the GUI. This determines the required inputs, i.e. ($\mathcal{W}$, P , $PROC$, $ROBOT$, C_t , $IsValid$).

2. Define number of path generation problems:

As defined in Subsection 3.1.1, the path generation for multiple selected product features can be treated with the single or the multiple problem strategy depending on the nature of the manufacturing process and if the costmap presents discontinuities. In the case just one feature is selected, the number of path generation problems is one, $\rho = 1$. If a single problem strategy is desired and multiple features are selected, then the piecewise approach is used and also $\rho = 1$ (refer to Subsection 3.1.1.1).

If the multiple problem strategy is selected (refer to Subsection 3.1.1.2), then $\varrho = u$. This will indicate the approach, that one iteration of the approach must be performed per feature. The multiple strategy permits to address discontinuities which appear on the RMPP^n due to the relation of the state cost and the features of the product. Example of this is the milling use-case in which orthogonal continuous features are treated (refer to Subsection 5.5.2). Here in synergy with the multiple problem definition, a heuristic algorithm and a genetic search have been applied in order to overcome the discontinuities. These are further detailed in step 5 of this approach.

3. Interpret the product and the process model:

The product (P) is interpreted taking into account the product features and the definitions of the translational and rotational components for defining the $T_w^p(k)$ as described in Subsection 3.1.1. Notice that each manufacturing process may have different translational and rotational models depending on its nature. For instance, in the welding use-case the optimal rotational component is parameterized with respect to k if multiple edges are involved (refer to Subsection 4.2.1). The process interpretation encodes the DoFs and also the constraints of the process in the transformation T_p^{proc} . The final interpretation is then the generation of equations defining the transformation T_w^{proc} as denoted in the previously introduced Equation 3.11.

4. Check the feasibility of the optimal path:

After determining the transformation from the world to the process coordinate frame T_w^{proc} , for some manufacturing cases, the optimal path can be determined without the need of exploring the DoFs ($\text{IsValid}(\cdot) \rightarrow \text{True} \vee \text{False}$). An example of this is the welding use-case, for which the optimal path is known, without knowing if this path is feasible through all the product features (refer to Chapter 4 for more detail about the welding process). Based on this, the feasibility can be evaluated to avoid sample-based planning in case of it not being required. For example, in the collision avoidance use-case, it is just required to perform all the interpretation and configuration of the sample-based algorithms in case a collision obliges the use of the DoFs. If the optimal or reference path is feasible, then no sample-based planning will be required.

5. Define the start and goal states:

Sample-based algorithms require the definition of start and goal states (s_s and s_g respectively). The software functionality $\text{StartGoalStates}(\cdot)$ is intended to define these for each process. Previous steps parameterize the path along the product features with the k dimension. Definition of the start and goal states are required for the other DoFs such as the rotational components which are not defined with respect to k . The start and goal states are in this approach minimized with respect to the specified state cost ($\mathbf{C}_s$) taking into account also the valid function ($\text{IsValid}(\cdot)$).

For defining the start and goal components in all DoFs, regions are defined by constraining $k_s = 0$ and $k_g = u \cdot k_d$ for the start and goal region respectively for the single problem strategy and with $k_s = (m - 1) \cdot k_d$, and $k_g = m \cdot k_d$ for each feature (m) with the multiple problem strategy and

by allowing exploration in the other DoFs. Remember that u is the number of selected continues features as defined in Subsection 3.1.1.

Heuristic and genetic search methods have been implemented for optimizing states in the start and goal region. For the welding use-case, a heuristic to find the range of collision is defined for finding the start and goal states (refer to Section 4.4). For the milling and sensing strategy optimization, the state cost function is minimized using the DoF in k_s and k_g with the genetic search algorithm. The workpiece position optimization algorithm in the welding and milling use-case also uses the genetic search for finding the minimum.

Moreover, if the multiple problem strategy is selected (refer to Subsection 3.1.1.2), the goal state of one feature and the start state of the next one in the path problem definition are constrained in $\mathbb{SE}^3$ with a heuristic algorithm to not deviate much in orientation and state cost in order to prevent robot poses deviating significantly in the Cartesian space.

6. Define the dimensions of the $\mathbb{RMP}^n$:

The path planning problem dimensions are required for establishing the relationship between the computed transformations and the functionalities on the overall operation of the approach. These dimensions are the DoFs of the PPR components. The function `DetermineDim (P, PROC)` defines the dimensions of the $\mathbb{RMP}^n$ ($DoF_{\mathbb{RMP}}$) by grouping the number of variables of each of the mathematical descriptions of the PPR components (i.e. DoF_p , DoF_{proc}) as expressed in Equation (3.14). The number of dimensions (n or $Dim_{\mathbb{RMP}}$) is also computed by performing the summation of the number of DoF of the product and process (i.e. Dim_p , Dim_{proc}) as expressed in equation (3.15). Figure 3.13 depicts the DoF of the product translation constraint parameterized with k and $\delta\alpha$ as example of a rotational DoF of the process. Besides, an extra dimension which defines the state cost is also illustrated. Note that this dimension is not part of the $\mathbb{RMP}^n$.

$$DoF_{\mathbb{RMP}} = DoF_p \cup DoF_{proc}, \quad (3.14)$$

$$n = Dim_{\mathbb{RMP}} = (Dim_p + Dim_{proc}) \in \mathbb{N} > 0. \quad (3.15)$$

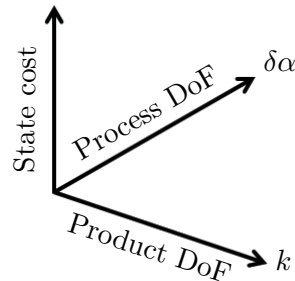


Figure 3.13: Representation of a $\mathbb{RMP}^2$ with extra dimension depicting the state cost.

7. Define bounds of the $\mathbb{RMP}^n$:

The bounds of the $\mathbb{RMP}^n$ are defined with the minimum and maximum value of the dimensions of the PPR components. These bounds are determined by the software function `Bounds (P, PROC)`.

Equation (3.16) describes for example the determination of the bounds for the k dimension with the single problem strategy and the rotational DoF $\delta\alpha$. Specification of the maximum and minimum values for the rotational component happens through the process model. The output of this function is the arrangement of vectors $\underline{B}$ of dimensions $[n \times 2]$ for this example. The first column is dedicated to the maximum values, the second one to the minimum ones. Each row references a dimension of the RMP. This example of bounds' settings is referred later in the welding use-case (refer to Section 4.4).

$$\underline{B} = \begin{cases} k, \begin{cases} k_s = 0 \\ k_g = u \cdot k_d \end{cases} \\ \delta\alpha, \begin{cases} \alpha_{min} \\ \alpha_{max} \end{cases} \\ \vdots \end{cases} \quad (3.16)$$

8. Set the $\mathbb{RMP}^n$ and compute the transformation $\mathbf{f}(\underline{s}) : \mathbb{RMP}^n \rightarrow \mathbb{SE}^3$:

In this step the $\mathbb{RMP}^n$ is set accordingly to the inputs of the PPR model and the definitions done from Step 5 to 7. As presented in Section 3.2, the $\mathbb{RMP}^n$ simplifies the path generation problem. Thus, for evaluating the feasibility or optimization, the mapping to $\mathbb{SE}^3$ is required. For achieving this, the mapping function is defined as follows,

$$\mathbf{f} \left(\begin{array}{c} DoF_p, \\ DoF_{proc}, \end{array} \right) : \mathbb{RMP}^n \rightarrow \mathbb{SE}^3 = T_w^p \left(DoF_p \right) \cdot T_p^{proc} \left(DoF_{proc} \right). \quad (3.17)$$

Where the DoF_p and the DoF_{proc} are the Degrees of Freedom (DoFs) of the product and the process, respectively. Note that this equation resembles the interpretation explained in the step 3 with the difference that this equation is directly associated to the C-space. Moreover, here all the related software functions are constrained to the dimensions of the C-space. For the studied use-cases, the DoF_p is defined by the k dimension and the DoF_{proc} by the free rotational components. Note that due to the modularity of the approach further extensions could be implemented. The computation of this function in software is denoted as $\text{RMPtoSE}^3(\cdot)$. This function is in charge of encoding the translation constraints of the products in the dimension k and setting the equations for the rotational DoFs.

9. Configure the path generation problem, sample, and solve:

Based on the previous steps, the $\mathbb{RMP}^n$ has been defined and the path planning problem and the Optimal Sample-based Algorithm (OSBA) can be configured. Then, the start and goal states, the total motion cost function, and the specification of the validity function are used for sampling and finding an optimized path σ^* with final cost C_t . This is denoted with the following equation,

$$\text{OSBA}(\mathbb{RMP}^n(\mathbf{f}, n, \underline{B}), s_s, s_g, \mathbf{C}_t, \text{IsValid}) \rightarrow (\sigma^*, C_t). \quad (3.18)$$

The resolution of the state validity checking function has been set to 0.001%. This resolution determines the discretization step of the added vertices to the path. The longest valid segment in a planning has been set higher than the maximum delta of bounds in order to allow that the planner finds solutions from the start to the goal state directly in case of this being possible. Some of the path planning algorithms require specific settings. This will be reported in this document when a planner which requires them is set.

Implemented and tested algorithms in the implementation chapter are: The RRT*, the T-RRT, the informed RRT*, and the Batch Informed Trees (BIT)* (see Subsection 2.1.3.4). A benchmark among these algorithms for the welding use-case is presented in Subsection 4.5.2.

10. Terminate automatically the path planning process:

Some of the optimal sample-based planning algorithms do not terminate automatically. An example of this is the RRT* which terminates planning after a certain amount of time specified by the end-user. In order to overcome this deficiency, a heuristic algorithm is implemented which uses as a basis the asymptotic behavior of the optimal sample-based algorithms for determining when an algorithm is coming close to its optimum. This heuristic algorithm is used in the approach in order to terminate automatically the optimization. This feature makes the configuration of the planning problem more intuitive for the end-user.

Algorithm 5 describes how the termination condition is conceptualized. The sample-based algorithm is started and the total motion cost is checked iteratively after each rewire loop. The rewire loop computes or tries to generate a new path (from start to goal) with new samples. A maximum planning time (t_{max}) is specified in order to terminate the algorithm in case the heuristic does not finish the process. If a better total motion cost is obtained, then the heuristic is triggered. A negative exponential function of the form $(a_i \cdot e^{-k_i \cdot t} + C_{\infty_i})$, where a_i , k_i , and C_{∞_i} are constants and t the elapsed time is fitted with the software function ($\text{Fit}(\cdot)$) having as inputs the tuples of values of the varied total motion cost and its corresponding time (C_t , and t respectively). In Subsection 4.5.2, final fittings for two plannings after finishing automatically the sample-based optimization for the welding use-case are depicted (Figures 4.10a and 4.10b).

Two rules are evaluated by the heuristic. The first one, evaluates the derivative of the motion cost w.r.t. the time evaluated at that time in which the last total motion cost was optimized. This value is compared with the desire delta of the derivative rule ΔD_{des} . Note that the ΔD_{des} is defined by the programmer after analyzing the exchange rate of the total motion cost with respect to the required planning time for the specific use-cases. This rule can be deactivated in the software, if desired. In Subsection 4.5.2, the derivatives of the motion cost for two specific path plannings are also depicted (Figures 4.11a and 4.12a). The second rule evaluates the delta between the fitted motion cost C_{∞_i} and the last computed total cost C_t with respect to the desired delta cost ΔC_{des} . In Subsection 4.5.2, the deltas of motion cost for two specific path plannings are also depicted (Figures 4.11b and 4.12b). The heuristic terminates the sample-based algorithm if each rule is satisfied in a number of repetitions (r).

Algorithm 5 Heuristic for the automatic termination of sample-based algorithms.

Input: OSBA, t_{max} , ΔC_{des} , ΔD_{des} , r , $i = 1$
Output: {stop sample-based algorithm}

```

1:  $t_i = 0$  {start timing}
2: while  $t_i < t_{max}$  do
3:    $(\sigma_i^*, C_{t_i}) \leftarrow \text{OSBA}(\mathbb{RMP}^n(\mathbf{f}, n, \underline{B}), s_s, s_g, \mathbf{C}_t, \text{IsValid})$  {check the optimization status of the OSBA}
4:   if  $(C_{t_i} < C_{t_{i-1}})$  then
5:      $(a_i \cdot e^{-k_i \cdot t} + C_{\infty_i}) \leftarrow \text{Fit}(\underline{C}_t, t)$  {fit equation}
6:      $D_i = \left( \frac{\delta C_t}{\delta t} \Big|_i < \Delta D_{des} \right)$  {heuristic rule 1: cost derivative w.r.t. time (true or false evaluation)}
7:      $\Delta C_{\infty_i} = ((C_{\infty_i} - C_{t_i}) < \Delta C_{des})$  {heuristic rule 2: delta of motion cost (true or false evaluation)}
8:     if  $(\Delta C_{\infty_i} \wedge \Delta C_{\infty_{i-1}} \cdots \wedge \Delta C_{\infty_{i-r}}) \wedge (D_i \wedge D_{i-1} \cdots \wedge D_{i-r})$  then
9:       return {stop sample-based algorithm}
10:    else
11:       $i = i + 1$ 
12:    end if
13:  end if
14: end while
15: return {stop sample-based algorithm}

```

11. Simplify or improve path:

After finding a path, the possibility of simplifying it by having less robot poses over it, discretizing it in a certain amount of poses, or to smooth the movements of the path is possible through algorithms. These functionalities are triggered automatically, if desired by the end-user, in order to enhance the final robotic path. In the implemented software, two simplification algorithms from the OMPL library are adopted: The interpolate function from the PathGeometric class and the reduce vertices function from the PathSimplifier class. These algorithms are available in the OMPL source code (Kavraki 2017).

Another path improvement algorithm has been implemented when the single problem strategy is addressed (refer to Subsection 3.1.1.1). Here the addition of robot states in the $\mathbb{RMP}^n$ is done when finishing or starting each product feature. This is done in order to not violate the translation constraints of the task, such as in the corners.

12. Retrieve the path to $\mathbb{SE}^3$ or $\mathbb{J}$ and generate a robot program:

The final step in the approach is to map the found feasible and optimized path from the $\mathbb{RMP}^n$ to the $\mathbb{SE}^3$ or $\mathbb{J}$. The mapping to $\mathbb{SE}^3$ is achieved using the automatically generated function $\mathbf{f}(\cdot)$. This step will not be denoted in the next presented algorithms. After retrieving the path, the simulation of the robot movement can be played in the software to visualize the results. The final step is to generate a specific robot program by parsing the path into the specific robot language such as the Kuka Robot Language (KRL) for the Kuka robots or the RSV for the Reis robots.

This section presented the proposed approach for solving the defined problem in Section 3.3. The following section presents an extension of this approach for computing close-to-optimal workpiece positions.

3.5 Approach Extension: The Automatic Close-to-Optimal Workpiece Positioning

To address improvement in the robotic manufacturing configurability and its programmability while maintaining manufacturing quality, the following path generation problem is additionally defined for demonstrating the usability of the proposed approach:

Compute automatically and offline a feasible and close-to-optimal workpiece pose T_w^{wp} (homogeneous transformation from the world (w) to the workpiece frame (wp)) in the Cartesian space ($\mathbb{SE}^3$) by exploring defined DoFs of the workpiece pose (position and/or orientation), optimizing the workpiece localization criteria $\mathbf{C}_{wp}$, and at the same time, the robotic manufacturing path (σ^*) under a specific total motion cost $\mathbf{C}_t$, weighted as defined in Equation (3.19). The weighting factors w_t and w_{wp} are defined by the end-user in order to specify the prioritization of the optimization.

$$\mathbf{C}_{t_{wp}} = w_t \cdot \mathbf{C}_t + w_{wp} \cdot \mathbf{C}_{wp} \quad (3.19)$$

In order to automatically solve the problem defined above, the solution approach previously proposed (refer to Section 3.4) is extended in order to include close-to-optimal workpiece positioning. For achieving this, first a systematic interpretation of the relations of the PPR components with respect to the path generation problem is addressed based on the specified manufacturing task. Second, the computation of an optimized path is performed using a sample-based algorithm by exploring the functional DoFs of the process (e.g. rotation around a milling tool or rotational DoFs for the welding process). Here if required, the exploration of the process constrained start and goal regions is performed for finding the optimal start and goal states in a certain workpiece position. Then, the planning is commanded by the workpiece positioning component which optimizes T_w^{wp} .

Finally, the workpiece position optimization component is used for approximating the global optimum over the costmap of the workpiece position DoFs on several iterations commanded by a simulated annealing algorithm. The objective of this process is to retrieve back a robot program with a close-to-optimal workpiece position with an optimized path over it. Figure 3.14 depicts the workflow of the close-to-optimal workpiece positioning in which the steps of each local path planning and the workpiece optimization are described as an extension of the general approach.

In order to optimize the workpiece pose, a simulated annealing algorithm is implemented to run the planner iteratively for finding a close-to-optimal pose. The termination condition is defined with an acceptance probability function that depends on the cost difference between the current and next state and the current temperature value of the simulated annealing algorithm. The algorithm searches different states using the Manhattan distance heuristic, which initially allows exploration in all directions to avoid getting stuck in a local minimum. The temperature of the algorithm decreases after each iteration in accordance with the defined cooling rate. After some time of exploration, a close-to-optimal solution is found T_w^{wp} and the optimized path σ^* is subsequently computed for that workpiece state, which is retrieved back as the solution.

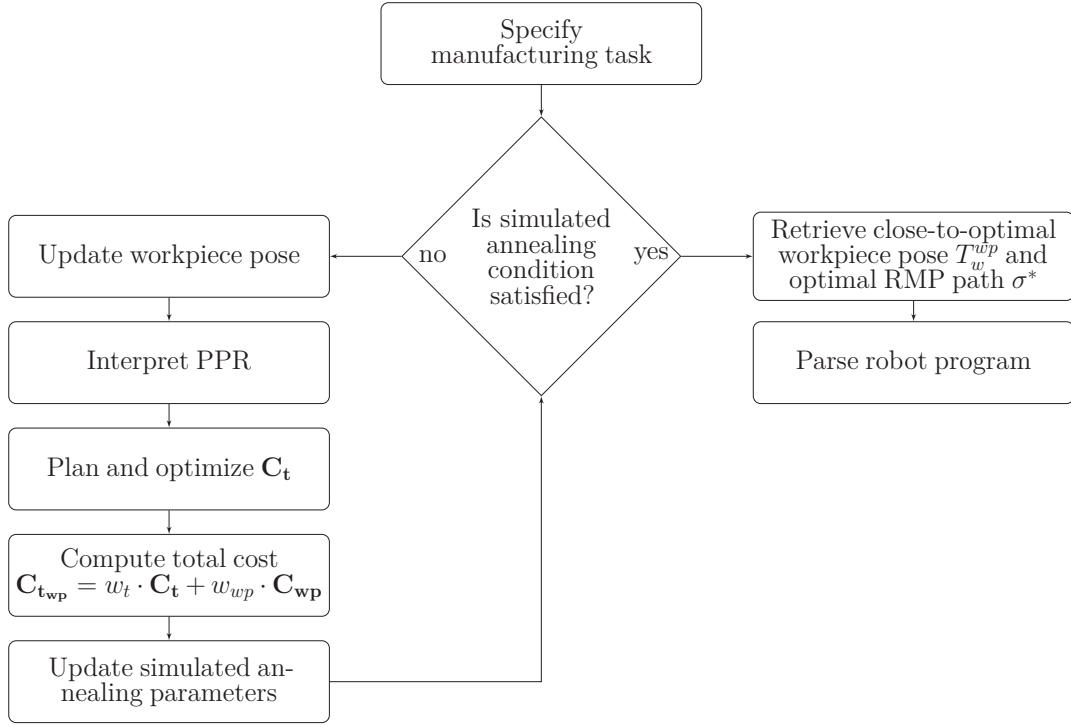


Figure 3.14: Flow diagram for the close-to-optimal workpiece positioning with optimized path planning.

3.6 Architecture and Software Implementation

The architecture of the AOLP approach is depicted in Figure 3.15. Inputs for the interpreter are the model-based descriptions of the PPR components as described in Section 3.1. Outputs of this interpreter are: (1) the configuration for the state validation function (refer to Subsection 3.2.1) which constitutes the formulation for determining if a state can be taken into account in the planning, (2) the manufacturing motion cost which constitutes a quantification of the relation between interconnected valid samples and the manufacturing optimum (refer to Subsection 3.1.2.2), and (3) the required configuration of a state-of-the-art optimal sample-based planning algorithm and the definition of the RMPⁿ.

The interpreter's outputs are then run into the sample-based algorithm for computing an optimized path which is later parsed into a robot program constituting the output of the proposed architecture and solution approach. If desired, workpiece pose optimization can be approached by iterating the planner as described in the previous section. The software implementation of this architecture is detailed below.

Software Implementation

An overview of the software implementation is depicted in Figure 3.17. The simulation environment runs in the programming language C#. CAD models can be loaded into the simulation environment by loading STLs and Object File Formats (OBJs). STEP models are used to generate the task description as described in Subsection 3.1.1. The Open Graphics Library (OpenGL) is used for manipulating the objects in the environment and constitutes the base for the scene visualization package. Initial positions and orientations of the objects are defined in the AML. The software has a GUI for allowing

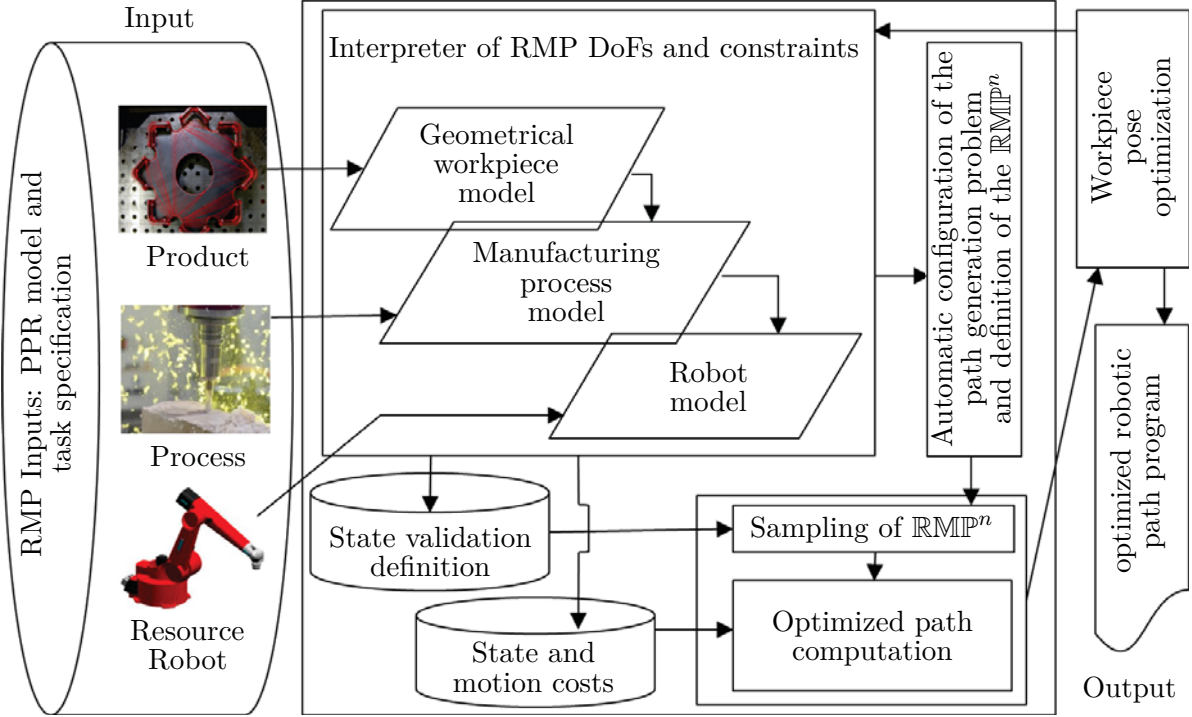


Figure 3.15: General architecture of the automatic optimal path generation for robotic manufacturing processes.

interaction with the end-user. Moreover, functionalities for selecting features over the workpieces in the simulation are also available.

In order to configure the AOLP system, the robot cell is described via AML using the PPR classification. This model-based description and software structure facilitates the changeover when facing different requirements of the PPR components. Different requirements are for instance, a different process, workpiece to be manufactured, robot provider, or the robot models to take into account, as shown in the use-cases described in Chapters 4, 5, and 6. The process model is defined in an XML format linked to the AML. This model specifies the geometrical relations between the product feature and the process frames as defined in Section 3.1. Moreover and depending on each process, specific parameters could be defined in this XML. Specific XML models are depicted in Subsections 4.2.2 and 5.2.2. Figure 3.16 depicts the first levels of a welding robot cell modeled in the AML editor, highlighting the components of the PPR model.

The robot models (refer to Subsection 3.1.3) such as the inverse and forward kinematics and the joint compliance model are defined in the C# environment. These models are also parameterized from the AML (e.g. the DH parameters). The main reason for implementing these models in the simulation environment is that they run simultaneously with the simulation. This assures a most efficient computation. Equivalently, the state validity functions (refer to Subsection 3.2.1) are also implemented in this environment because they relate the objects of the scene and the robot models.

The product models for the translational and rotational features defined in Subsection 3.1.1 are coded in Python. A software module for interpreting the geometrical process specifications given in the XML format is also conceived in this programming language. These models are defined as mathematical equations in a symbolic way using the SymPy library. Mathematical equations are

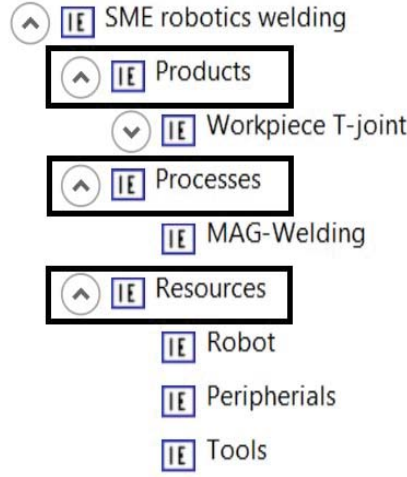


Figure 3.16: Model-based description in AML using the PPR model categorization.

automatically generated based on the end-user input taking into account the specified transformations (refer to Section 3.1). This step is illustrated in Figure 3.17 with the software functionality "RMP transformation equations generation" which also includes the algorithms for generating the piecewise equations, previously defined in Subsection 3.1.1, in case the RMP task specification demands it.

The main advantage of this implementation is that mathematical descriptions are automatically determined depending on the required configuration. For interfacing these equations with the sample-based algorithm, a C file is automatically generated evoking the symbolic equations for being compiled into a Dynamic-Link Library (DLL) using the GNU Compiler Collection (GCC) or the Microsoft Visual Studio 2010 compiler (cl.exe).

The state and motion cost models are defined in C++. They are referenced from the XML and are used in the interpretation step. After the interpretation (refer to Section 3.4), a sample-based algorithm is configured in OMPL. The OMPL version linked as a library is the 1.1.0.

Furthermore, an automatic algorithm for defining start and goal regions constrained in the k dimension (k_s and k_g , start and end point of the parameterized product feature) is also implemented using the goalregion class of the OMPL (Kavraki 2017). A genetic search algorithm, using the OMPL GeneticSearch class, is embedded in C++ in order to find the most optimal start and goal state while avoiding the infeasibilities. Heuristic algorithms are also implemented to find the optimal start and goal states.

The path generation, programmed in C++, is compiled into an DLL linked to the simulation software. The output of the architecture after the input of the end-user, which is to define the task, is a path in the RMP C-space which is transformed to the Cartesian or joint space for later simulating or parsing it into a robot program.

The implementations performed on the framework of this dissertation are all the modules specified under the optimal path generation, the robot modeling, the implementation of the validity functions and the required adaptations to the AML and XML models in order to interface them with the planning approach and architecture. All the computations presented in the implementation chapters (from 4 to 6) have been performed with the same hardware¹. All the coordinate systems depicted in these

¹Computations were performed in a Dell Precision M4800 laptop with processor Intel Core i7-4900MQ - 2.8 Hz and 16 Gb of RAM

chapters follow the Red Green Blue (RGB) convention- red, green, blue for $\vec{x}$, $\vec{y}$ and $\vec{z}$ respectively-. Figure 3.17 depicts the overview of the previously described software implementation.

3.7 Conclusion

This chapter presented the model-based description of Robotic Manufacturing Processes (RMPs). The product models use CAD files of the workpiece into the simulation environment. As further modeling, equations constraining the translation and optimal or desired orientations with respect to the product are generated automatically based on the model-based description of the features. The manufacturing process is modeled geometrically with the homogeneous transformation from the product to the process coordinate system by defining the DoFs and constraints of it. These specifications are set via an XML file. Moreover, the process is also defined with state and motion cost which define the optimality of the manufacturing task. The robot model is adopted from the state of the art and configured via an AML describing the robot cell.

The robotic manufacturing configuration space ($\mathbb{RMP}^n$) is defined as the C-space in which the product translational constraints and the optimal orientations through a product feature are set in one dimension and the further DoFs of the process as further dimensions. The maximal dimension of this space in the implementation is four, one dimension constraining the product features and three process rotational DoFs. Nevertheless, this space can be further extended to any dimensionality as required. The $\mathbb{RMP}^n$ allows and simplifies the computation of optimized paths for robotic manufacturing using sample-based algorithms. Furthermore, the problem for planning optimized paths for robotic manufacturing process is defined and solved with the proposed approach and architecture.

The proposed approach, named the Automatic Optimized Offline Programming (AOLP), consists of three main steps, namely (1) the systematic interpretation of the manufacturing process using as inputs the proposed model-based description of the PPR, (2) the automatic configuration of a state-of-the-art optimal sample-based algorithm and a path planning problem in the interpreted $\mathbb{RMP}^n$ and (3) the sampling and solving in the $\mathbb{RMP}^n$, for finally generating an optimized robot program. The architecture, describing also the software implementation, has been detailed in this chapter.

This approach and architecture are proposed for solving the stated research question: How to automatically generate offline optimized paths in reaction to infeasibilities for robotic manufacturing processes under constraints and process optimization criteria? Ultimate goals of the approach and architecture are to improve the required programming time and the quality of robotic manufacturing processes. Evaluation of these goals is performed in the proposed use-cases.

The next three chapters present the implemented and evaluated use-cases. The definitions of the use-cases based on the deficiencies encountered in the state of the art and their analysis were performed in Subsection 2.7.2. For each use-case different product, process, and resource models are taken into account. Notwithstanding, the same architecture and approach have been used to solve the path optimization problem and some models have been used in multiple use-cases. The use-cases are: (1) the path generation for the optimized collision avoidance in robotic welding (refer to Chapter 4), (2) the optimized stiffness in robotic milling (refer to Chapter 5), and (3) the optimized sensing strategy for robotic deburring (refer to Chapter 6).

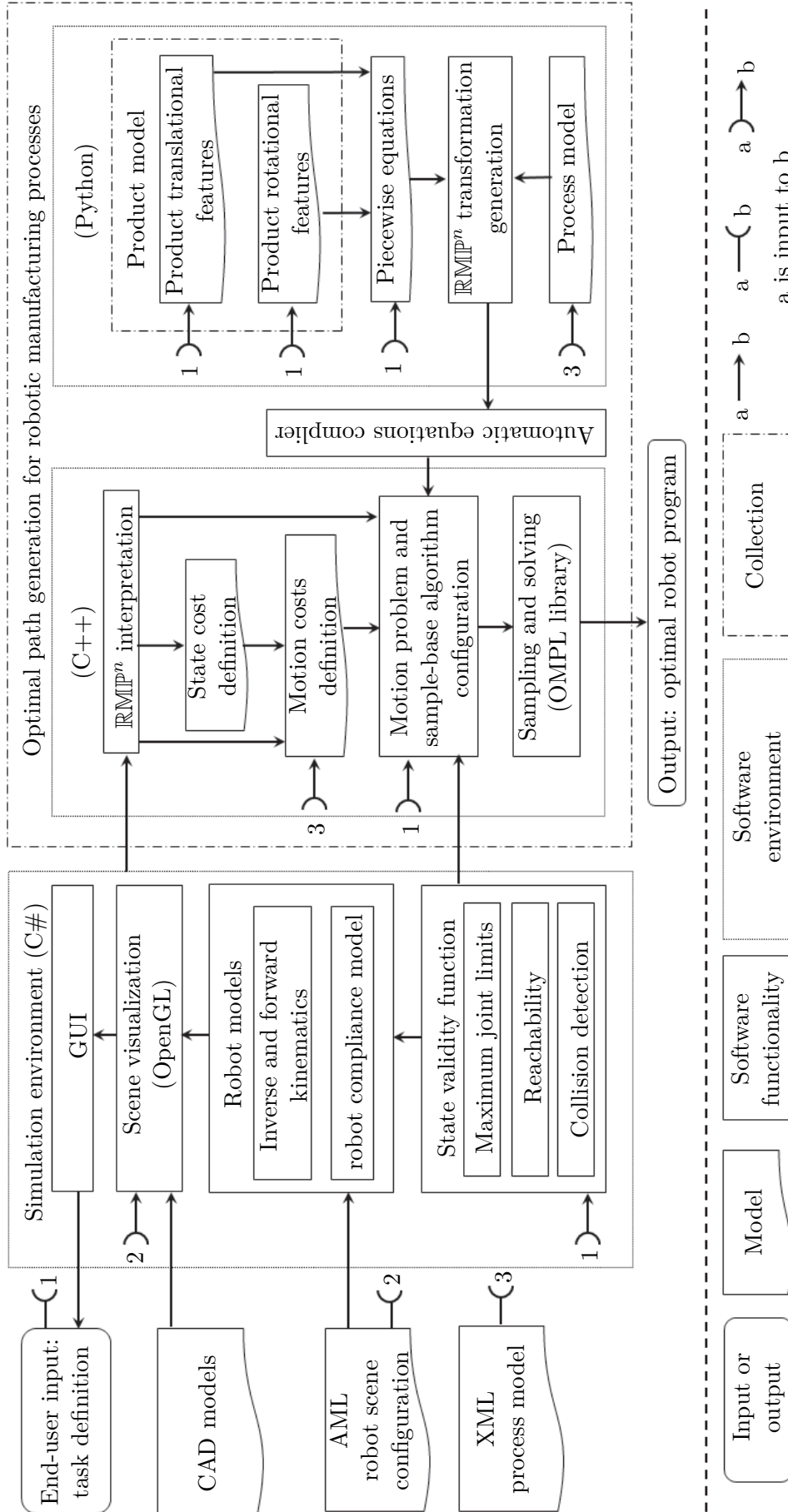


Figure 3.17: Overview of the software implementation.

4 Optimized Collision Avoidance in Robotic Welding

This chapter presents the implementation, experimental investigation, and results of the welding use-case. The problematic and motivation of this use-case was defined in Subsection 2.7.2. This chapter is organized as follows: Section 4.1 defines formally the problem of optimized collision avoidance in robotic welding. Section 4.2 describes the model-based structure of this use-case. Section 4.3 introduces the state and motion cost. Section 4.4 describes the interpretation of the robotic welding process and the automatic configuration of optimal sample-based algorithms for the computation of optimized paths when collision avoidance is required. Section 4.5 presents the simulation and the experimentation. Section 4.6 concludes this use-case.

4.1 Problem Definition

The following problem is defined: compute automatically and offline an optimized collision-free path σ^* satisfying the product and process constraints. The robot must optimize the welding torch orientation with respect to the welding process optimal orientation. The optimization is enabled by using the degrees of freedom of the process throughout all the welding seam while avoiding the collisions encountered (Diaz Posada, Dietz et al. 2016).

4.2 Product, Process, and Resource Modeling

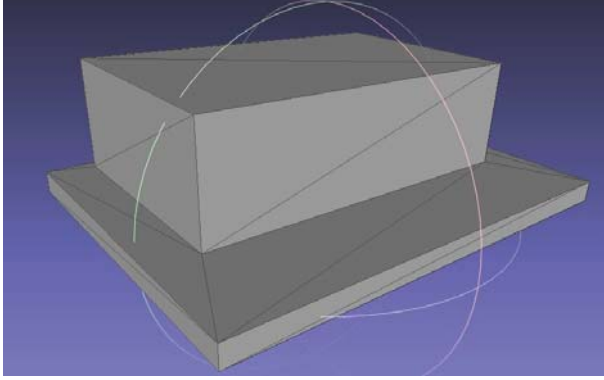
This section describes the proposed PPR model representation for this use-case.

4.2.1 Product Model

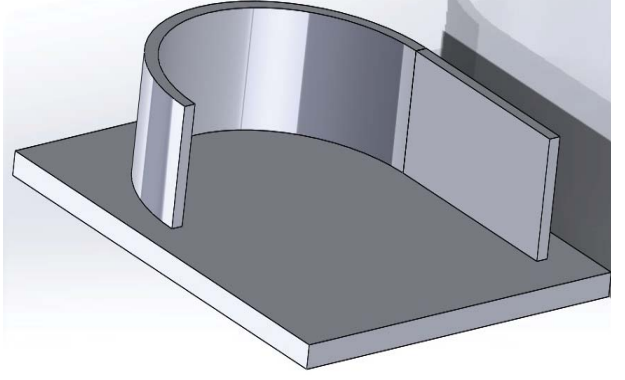
The welding product model, denoted as (P) and generally described in Subsection 3.1.1, consists of the mathematical description using homogeneous transformations of single or multiple continuous T-joint seams of a welding workpiece. These weld seams are selected by the end-user by clicking in the simulation over the workpiece STL model.

Two different workpieces will be used in this use-case. The simulation and experimentation results are presented for each workpiece: The first one is composed of just linear features (refer to Figure 4.1a). This workpiece is addressed in order to demonstrate the generation of piecewise equations with the single problem strategy used to deal with continuous product geometrical features. For this workpiece, a specific model in which the optimal rotation is interpolated between features' intersections is presented, this being a typical welding requirement. The description of the translational component of the homogeneous matrix consists of piecewise equations for this workpiece as defined in Subsection 3.1.1.

The second workpiece is composed of a semi-circumferential and a linear feature for demonstrating that the approach can interpret different geometrical descriptions (refer to Figure 4.1b). For the second workpiece, the multiple problem strategy is selected.



(a) Model of a workpiece with linear features.



(b) Model of a workpiece with a semi-circumferential and a linear feature.

Figure 4.1: Product models of workpieces for the welding use-case.

The rotational component of the welding product is modeled based on a defined orientation convention for the T-joint seams. A quaternion, denoted as $\underline{q}_m$, is computed based on the normal vectors of the two T-joint planes for linear features. Remember that m identifies one of the selected product features as defined in Subsection 3.1.1. For curved features the tangential plane is taken into account. Note that this convention can be defined as desired. However, the process model must be parameterized based on this definition. A percentage of the k parameterization (as defined in Subsection 3.1.1) in which the rotation between seams may occur is introduced and denoted as ζ (refer to Figure 4.2). ζ is then specified between 0 and 100. Rotations from seam (m) to seam ($m + 1$) are calculated as follows,

$$\delta \underline{q}_{([m] \rightarrow [m+1])} = (\underline{q}_m)^{-1} \cdot \underline{q}_{[m+1]}, \quad (4.1)$$

and transformed to the axis-angle representation (θ_m for the angle and $\vec{T}_m$ for the vector) with the following function

$$[\theta_m, \vec{T}_m] = \mathbf{Q2AA}(\delta \underline{q}_{([m] \rightarrow [m+1])}). \quad (4.2)$$

Based on the computation of θ_m and the definition of k_d which is the end of parametrization of a single product feature set as 100 as defined in Subsection 3.1.1, the optimal rotation from one weld seam to the next continuous weld seam is defined for the ζ percentage of each weld seam parameterizing this in k with the angle to rotate around $\vec{T}_m$ as follows,

$$\mathbf{g}_m(k) = \theta_m \cdot \left(\frac{k - (k_d \cdot m - \zeta)}{2\zeta} \right). \quad (4.3)$$

This rotational component is defined for the number of selected seams with optimal rotation interpolation between feature's intersections as described in Lines 7 and 8 of Algorithm 6 for the first workpiece. Algorithm 6 describes the automatic interpretation of the multiple seam product model with optimal rotation interpolation between feature's intersections with the single problem strategy

For the second workpiece (the one with the semi-circumference), the optimal rotation interpolation between feature's intersections is not required because the rotations between the product features is zero. Therefore, Equation (4.3) is not assigned in this case and instead the quaternion defining the product is assigned as the optimal orientation.

Algorithm 6 uses the function $\mathbf{R}_q(q_m)$ to calculate a rotation matrix from a quaternion and the function $\mathbf{R}_{aa}(\mathbf{g}_m(k), T_m)$ for transforming from the axis-angle into the rotational matrix notation. The descriptions of the rotation and the translation are then used as in Algorithm 6 for describing the homogeneous matrix for each range. Remember that u denotes the number of selected continues features. $\vec{V}_{sm}$ and $\vec{V}_{gm}$ denote the start and goal position vectors, respectively, of a product feature m as defined in Subsection 3.1.1.

The output of Algorithm 6 is the transformation world to product $T_w^p(k)$ parameterized in k , constraining the product translational features and defining the optimal orientation, as described in Equation (4.4). Note that this equation is automatically generated by the algorithm taking into account the product model. The dimension of the $\mathbb{RMP}^n$ determining the product translational constraints and optimal orientation is then determined with the variable k (denoted also as DoF_p). Figure 4.2 depicts the geometrical descriptions previously introduced.

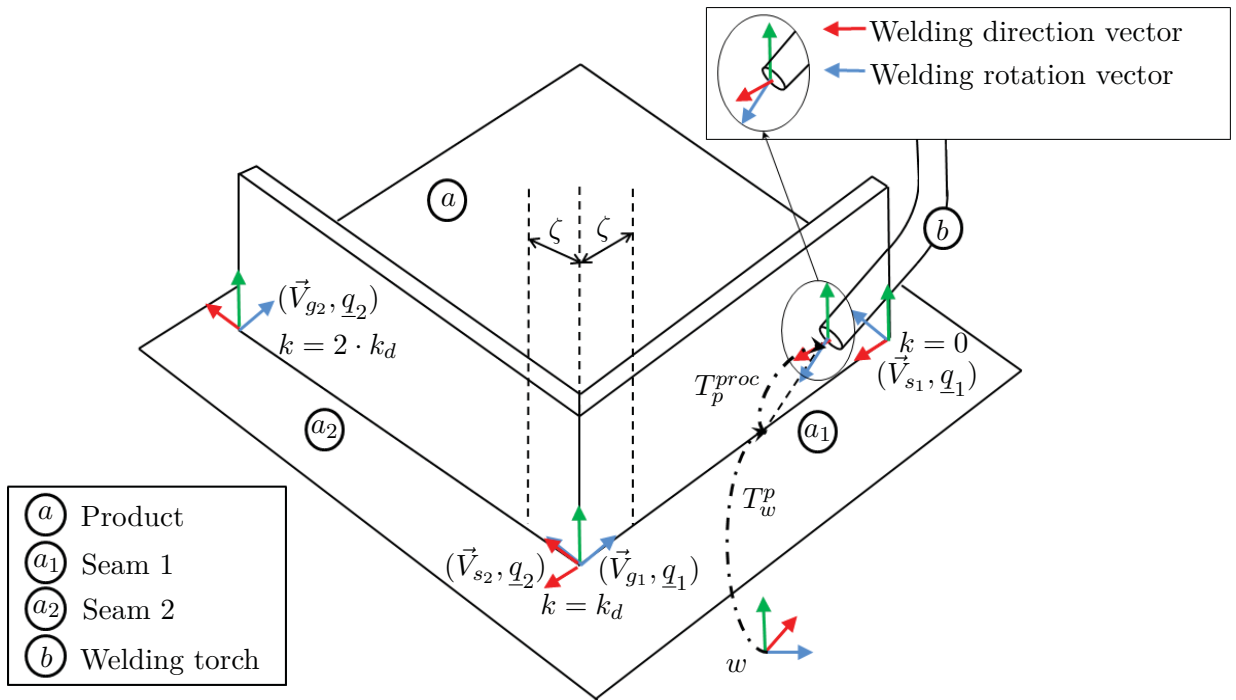


Figure 4.2: Illustration of two continuous T-joint seam product models and their notations.

Algorithm 6 Algorithm for automatic interpretation of the multiple seam product model with optimal rotation interpolation between feature's intersections with the single problem strategy.

Input: $(\vec{V}_{s_1} \dots \vec{V}_{s_u}), (\vec{V}_{g_1} \dots \vec{V}_{g_u}), (\underline{q}_1 \dots \underline{q}_u), (\vec{T}_1 \dots \vec{T}_u), \zeta, u, k_d$

Output: $T_w^p(k)$ {automatic generated equation} (4.4)

```

1:  $T_w^p(\text{DoF}_p) = T_w^p(k) =$ 
2: if  $u = 1$  then
3:    $\begin{bmatrix} \mathbf{R}_q(\underline{q}_1) & \vec{V}_{s_1} + (\frac{k}{k_d})(\vec{V}_{g_1} - \vec{V}_{s_1}) \\ 0 & 1 \end{bmatrix}, k \in [0, k_d]$ 
4: else
5:    $\begin{bmatrix} \mathbf{R}_q(\underline{q}_1) & \vec{V}_{s_1} + (\frac{k}{k_d})(\vec{V}_{g_1} - \vec{V}_{s_1}) \\ 0 & 1 \end{bmatrix}, k \in [0, (k_d - \zeta))$ 
6:   for  $m = 1$  to  $(u - 1)$  do
7:      $\begin{bmatrix} \mathbf{R}_q(\underline{q}_m) \cdot \mathbf{R}_{aa}(\mathbf{g}_m(k), \vec{T}_m) & \vec{V}_{s_m} + (\frac{k}{k_d})(\vec{V}_{g_m} - \vec{V}_{s_m}) \\ 0 & 1 \end{bmatrix}, k \in [(k_d \cdot m - \zeta), (k_d \cdot m))$ 
8:      $\begin{bmatrix} \mathbf{R}_q(\underline{q}_{m+1}) \cdot \mathbf{R}_{aa}(\mathbf{g}_m(k), \vec{T}_m) & \vec{V}_{s_{m+1}} + (\frac{k}{k_d})(\vec{V}_{g_{m+1}} - \vec{V}_{s_{m+1}}) \\ 0 & 1 \end{bmatrix}, k \in [k_d \cdot m, (k_d \cdot m + \zeta))$ 
9:      $\begin{bmatrix} \mathbf{R}_q(\underline{q}_{m+1}) & \vec{V}_{s_{m+1}} + (\frac{k}{k_d})(\vec{V}_{g_{m+1}} - \vec{V}_{s_{m+1}}) \\ 0 & 1 \end{bmatrix}, k \in ((k_d \cdot m + \zeta), (k_d(m + 1) - \zeta)]$ 
10:     $m = m + 1$ 
11:   end for
12:    $\begin{bmatrix} \mathbf{R}_q(\underline{q}_u) & \vec{V}_{s_u} + (\frac{k}{k_d})(\vec{V}_{g_u} - \vec{V}_{s_u}) \\ 0 & 1 \end{bmatrix}, k \in ((k_d \cdot u - \zeta), k_d \cdot u]$ 
13: end if
14: return  $T_w^p(k)$ 

```

4.2.2 Process Model

Position and rotation constants of the transformation T_p^{proc} , as defined in Subsection 3.1.2, are values derived from the standardization or from the expertise of welders. For T-joints which are the product features treated in this use-case, an optimal welding angle of half the angle between both T-joint planes is recommended. The $\vec{x}$ direction of the process frame defines the welding path. The $\vec{z}$ vector defines the vector in which the welding torch points. The DoFs can be defined as any of the rotational components. As first example, the DoF is defined as $\delta\alpha$, being this the rotation around $\vec{x}$.

The process is modeled into an XML file linked to the AML as defined in Subsection 3.1.2. Listing 4.1 shows an XML model with the characteristics previously defined activating one rotational DoF. Dark blue texts represent software identifiers and black texts are characteristics defined by the end-user. This model also defines a fixed value in the $\vec{z}$ component, being this the stick-out required by the welding process. The XML describes the geometrical transformation from product to process T_p^{proc} (depicted in Figure 4.2). Parameters defined under the semantic specification (`<FrameConstraintsDoF>`) specifies the constraints or fixed dimensions (`type="Fixed"`) and the DoFs of the manufacturing process (`type="Range"`). The minimum (`min=""`) and maximum (`max=""`) values (α_{min} and α_{max} as example for the $\delta\alpha$ DoF) are assigned as bounds as presented in Equation (3.16). Constant values are defined under the category (`value=""`). These values are linked in the path planner to Equation (3.6). The

models for the optimization criteria, defined in Section 4.3, for this use-case are identified in the software under the semantic name specified in the category (`OptimizationFunction name=""`).

```
1<Process name="Welding" type="T-Weld">
2  <FrameConstraintsDoF>
3    <GeoParameter name="x" unit="mm" type="Fixed" value="0"/>
4    <GeoParameter name="y" unit="mm" type="Fixed" value="0"/>
5    <GeoParameter name="z" unit="mm" type="Fixed" value="10"/>
6    <GeoParameter name="α" unit="Deg" type="Range" min="-20" max="30" value="45"/>
7    <GeoParameter name="β" unit="Deg" type="Fixed" value="0"/>
8    <GeoParameter name="γ" unit="Deg" type="Fixed" value="0"/>
9  </FrameConstraintsDoF>
10 <OptimizationFunction name="Optimized_Collision_Avoidance_Welding">
11 </OptimizationFunction>
12</Process>
```

Listing 4.1: Semantic modeling of the welding process in XML format

4.2.3 Robot Model

The resource model, denoted as (ROBOT), in this case an industrial robot, is defined with the forward and inverse kinematics as defined in Subsection 3.1.3 which allows the mapping from $\mathbb{J}$ to $\mathbb{SE}^3$, and the other way around respectively, as defined in Section 3.2.

4.3 State and Motion Cost

4.3.1 State Cost

The welding process is defined as optimal if no deviation in orientation is required from the optimal rotation (fixed values of the process model). This optimal rotation is defined by process experts, standards and/or technical experimentation. Based on the definitions of Section 4.2, the welding process is optimal if the process DoFs are zero, meaning that the optimal transformation from world to process is described as follows,

$$T_w^{proc_{opt}}(k) = T_w^p(k) \cdot T_p^{proc}(0). \quad (4.5)$$

Based on this optimal function for the welding process, the state cost is defined as the minimum required angle (θ_{s_i}), calculated using the axis-angle representation, of a non-collision generated sample ($s_i = [k_{s_i}, \alpha_{s_i}]$ as example for one rotational DoF) for reaching the optimum, defined as $\theta_{opt} = 0$, on a certain k value of the parameterized T-joint seams. The state cost is denoted as follows,

$$\mathbf{C}_s(s_i) = |\theta_{s_i}|. \quad (4.6)$$

Note that the state cost is one dimensional and independent of the number of rotational DoFs. It is calculated based on the orientation in $\mathbb{SE}^3$ which is mapped from the generated sample in the $\mathbb{RMP}^n$. The notation θ_{s_i} represents the minimum required angle of rotation to reach the optimum orientation.

4.3.2 Motion Cost

The manufacturing motion cost $\mathbf{C}_M(\cdot)$ for this use-case is defined as the integral function of the state cost of the interconnected states through all the path as follows,

$$\mathbf{C}_M(\sigma) = \int \mathbf{C}_s dk. \quad (4.7)$$

Note that the definition of this motion cost is valid for any path problem dimensionality. As example for one rotational process DoF ($s_1 = [k_{s1}, \alpha_{s1}]$, and $s_2 = [k_{s2}, \alpha_{s2}]$), the manufacturing motion cost is calculated by computing the area under the costmap which connects the minimum angle of rotation of these two samples (θ_{s1} and θ_{s2} for sample 1 and 2, respectively). For two rotational DoFs, the integral is computed as the volume below the costmap.

Moreover, the motion cost policy $\mathbf{C}_p$, introduced in Subsection 3.1.2.2 in order to avoid backward motions which are undesired for the welding process, is applied in this use-case. The total motion cost is then calculated as the sum of the manufacturing motion cost and the backward cost policy in case the angular velocity motion cost is not selected (refer to Subsection 3.1.2.2 and to Subsection 4.5.2 for the later use of this cost in this use-case). The total motion cost for the welding process is then defined as follows,

$$\mathbf{C}_t = \mathbf{C}_M + \mathbf{C}_p. \quad (4.8)$$

4.4 Optimized Path Generation for Collision Avoidance in Robotic Welding

This section describes how the proposed approach, presented in Section 3.4, interprets the welding process and configures the sample-based algorithm for finding an optimized path for avoiding collisions while maintaining the optimum welding angle as much as possible. Note that in this section just the particularities of this use-case in respect to the general solution approach are presented.

The product, process, and resource models for this use-case were defined above. Specific product models were implemented for defining the optimal rotation when dealing with continuous multiple weld seams (refer to Subsection 4.2.1). Moreover, the state and total motion cost for the robotic welding were also specified in Section 4.3. These components and the specification of the validity function, in this use-case the collision detection (refer to Subsection 3.2.1), compose the required input definition and allow the definition of the transformation T_w^{proc} .

A particularity of this use-case is that the optimal path is known beforehand as defined in Equation (4.5). Based on this definition, it is determined in the simulation environment if the optimal path ($T_w^{proc_{opt}}$) is practicable by evaluating the feasibility of this transformation in the range of $[0, k_g]$ by using the function $\text{IsValid}(\cdot)$ as follows,

$$\text{True} \vee \text{False} \leftarrow \text{IsValid} (T_w^{proc_{opt}}([0, k_g])). \quad (4.9)$$

If the optimal path is viable, the implemented software returns the path which is also a useful functionality for the programming experience. However, this work focuses on optimal reactions to infeasibilities such as collisions which could appear on the path and which requires the robot manipulator to rotate in order to avoid them. In case of collision if desired by the end-user, a function determines the range of the feature in which the collisions are present. The function uses the $\text{IsValid}(\cdot)$ and the

state cost $\mathbf{C}_s(\cdot)$ evaluations as an extension to reduce the search in the $\mathbb{RMP}^n$. This function, named $\text{CollisionRange}(\mathbf{P}, \text{PROC}, \text{ROBOT}, \mathbf{C}_s, \text{IsValid})$, determines for this use-case the start s_s and goal state s_g parameterized in k before and after the collision taking into account a specific motion percentage among k , defined by the end-user. The values in the k dimension of these states ks_s , ks_g are used for defining the ranges of the optimal solution. The paths before ks_s and after ks_g , σ_{j-1}^* and σ_{j+1}^* respectively, are then defined as optimal and are not interpreted for further sample-based optimization.

Note that in this specific use-case, if the range is automatically determined, the start and goal state in the k dimension are not defined as $k_s = 0$ and $k_g = k_d$ as in the general approach. If the determination of the collision range is desired, the general function $\text{StartGoalStates}(\cdot)$ is substituted by the function $\text{CollisionRange}(\cdot)$. Other aspects of the general approach remain the same. Thus, the dimensions of the process are defined as described in Section 3.4. Note that due to the general proposed approach just specific changes related to the nature of the process are required. These changes are commanded from the process descriptor given in the XML.

Finally after the interpretation, the robotic manufacturing process is configured into a state-of-the-art sample-based path generation algorithm. Notice that for the welding use-case and the first workpiece, the single problem strategy is addressed ($\rho = 1$), because it facilitates a faster planning over the welding costmap that does not present discontinuities. The reason for this is that the rotational component has been parameterized in relation to the dimension k . Meaning that the $\mathbb{RMP}^n$ does not present discontinuities. Note that this is not the situation for the later presented use-cases. For this use-case, the results in the next section will be presented using the RRT* algorithm. Algorithm 7 presents the approach for finding the optimized motion σ^* taking into account the presented models and the general approach definition given in Section 3.4.

Algorithm 7 Automatic interpretation and calculation of optimized collision avoidance for robotic welding using RRT*.

Input: $\mathbf{P}$ (eq. (4.4)), PROC (refer to Subsection 4.2.2), ROBOT (as defined in 4.2.3), $\mathbf{C}_t$ (eq. 4.8), IsValid (refer to Subsection 3.2.1), k_g

Output: σ^* {optimized path avoiding the collision}

```

1:  $T_w^{proc_{opt}}(k) \leftarrow \text{Interpret}(\mathbf{P}, \text{PROC}, \text{ROBOT})$  (as in eq. (4.5))
2: if  $\text{False} \leftarrow \text{IsValid}(T_w^{proc_{opt}}([0, k_g]))$  then
3:    $\mathbf{f} \leftarrow \text{RMP}_{\text{toSE}^3}(\mathbf{P}, \text{PROC}, \text{ROBOT})$ 
4:    $[s_s, s_g] \leftarrow \text{StartGoalStates}(\mathbf{P}, \text{PROC}, \text{ROBOT}, \mathbf{C}_s, \text{IsValid}) = \text{CollisionRange}(\cdot)$ 
5:    $\underline{B} \leftarrow \text{Bounds}(\mathbf{P}, \text{PROC})$ 
6:    $n \leftarrow \text{DetermineDim}(\mathbf{P}, \text{PROC})$ 
7:   {sample and calculate optimized path over the collision}
    $\sigma_j^* \in [ks_s, ks_g] \leftarrow \text{RRT}^*(\mathbb{RMP}^n(\mathbf{f}, n, \underline{B}) s_s, s_g, \mathbf{C}_t, \text{IsValid})$ .
8:   {retrieve optimal paths without collision}
    $\sigma_{j-1}^* \in [0, ks_s]$  and  $\sigma_{j+1}^* \in [ks_g, k_g]$ 
9:   return  $\sigma^* = \{\sigma_{j-1}^*, \sigma_j^*, \sigma_{j+1}^*\}$  {concatenation of optimal and optimized paths}
10: else
11:   return  $\sigma^* = T_w^{proc_{opt}}(k) \in [0, k_g]$ 
12: end if
```

4.5 Optimization in OLP Software and Experimentation

To evaluate the proposed approach and architecture, two different workpieces and different collision configurations for robotic welding are tested. The workpieces are addressed for demonstrating the product features modeling. The workpieces were explained and depicted in Subsection 4.2.1.

4.5.1 Experimental Setup

The welding robot cell modeled in the simulation environment and used for the experimentation of this use-case consists of an industrial robot Reis RV30-26 provided with a welding rotatory table and a Schunk tool changer reference SWK-040Q. The robot cell is provided with two end-effectors. One end-effector has a 2D laser scanner reference scanControl 2900-50 from the company Micro-Epsilon. In the other, the welding torch is mounted. Figure 4.3 depicts the welding robot cell and its components. A TCP/IP connection is available to communicate the simulation environment with the robot controller in order to transfer automatically programs generated with the previously described interpretation and configuration of the RRT* algorithm.

The robot cell is simulated using the CAD files of its elements into the internally developed software configured with the PPR model-based characterization described into the AML format as introduced in Section 3.6. The robot, peripherals, and end-effectors are described under this resource classification. The CAD files of the Reis robot joints and its Denavit-Hartenberg kinematic parameters are associated with the resource structure. The CAD file of the product (workpiece) is also linked in this AML format.

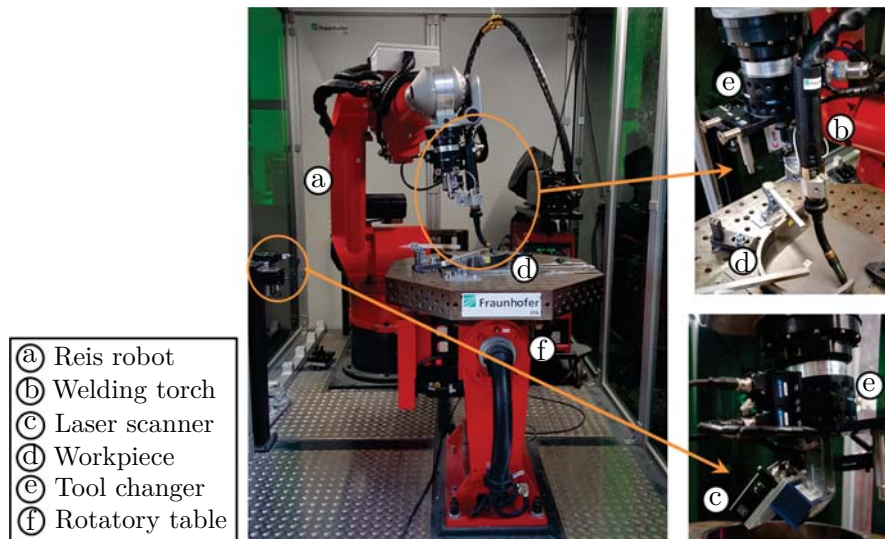


Figure 4.3: Welding robot cell.

4.5.2 Optimization in OLP Software

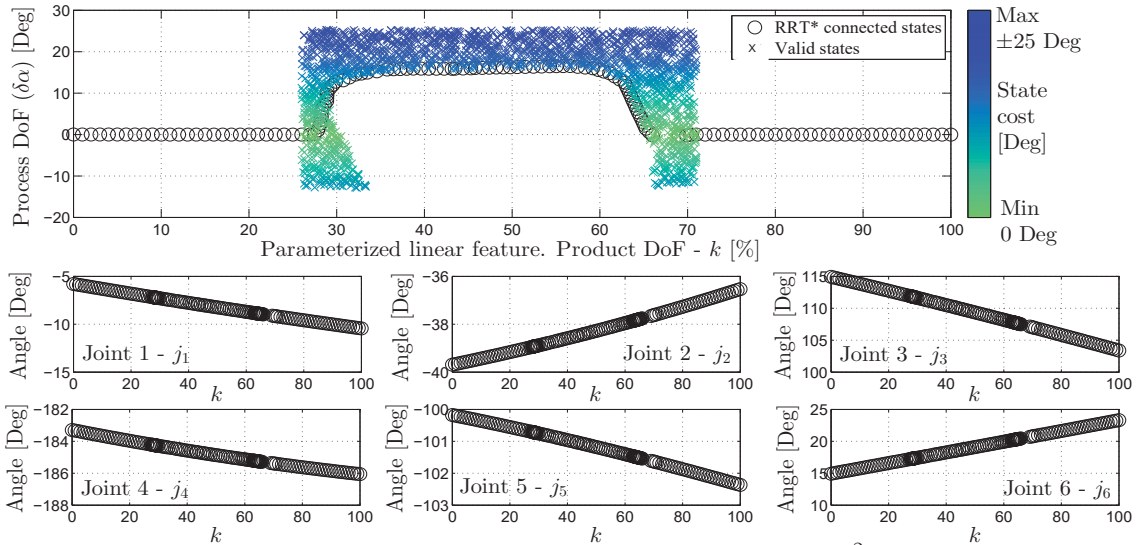
The following subsection presents the results obtained for the two different analyzed workpieces.

Workpiece with linear features

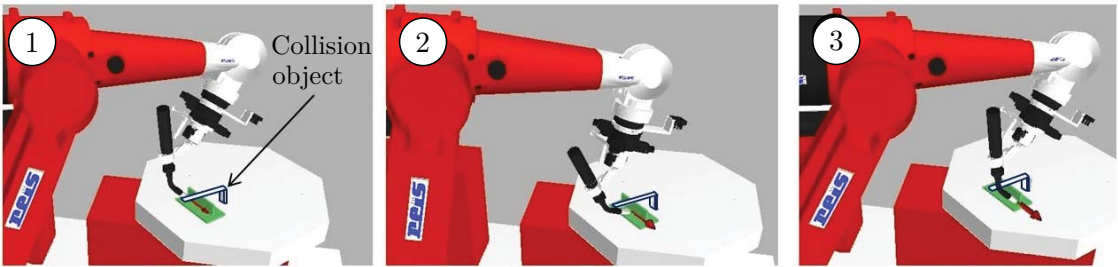
The first workpiece evaluates a collision avoidance for a T-joint seam of linear components in three configurations in simulation (Diaz Posada, Dietz et al. 2016). A fixed rotation of +45 Deg is set from the product orientation in $\delta\alpha$ and the stickout is defined for this workpiece as 10 mm for the three

configurations as in listing 4.1 in page 84. The first configuration set the rotation around the welding direction vector (i.e. $\delta\alpha$) as DoF of the manufacturing process as displayed in listing 4.1. Maximum and minimum bounds for this DoF are +30 and -20 Deg respectively. In the second configuration the rotation around the welding rotation vector (i.e. $\delta\gamma$) is set as DoF. Maximum and minimum bounds for this DoF are defined as +50 and -150 Deg respectively. A change of the DoF is addressed to illustrate the functioning of the XML parametrization. And for the third, a collision avoidance over the intersection of two product features is evaluated for demonstrating the advantage of the single problem strategy with the algorithm for generating piecewise equations as introduced in Section 3.4.

After the automatic configuration presented in Subsection 4.4, the RRT* finds an optimized path for avoiding the collision using $\delta\alpha$ as depicted in Figure 4.4a. The collision object during the welding is an L form bar depicted in Figure 4.4b. The results in joint space and a sequence of robot movements are also depicted. The sample cost of each valid sample is observed thanks to the color map. The right part and left part of the motion were not sampled due to the pre-definition of optimum for this specific manufacturing process using the function for finding the start and goal states. Figure 4.4a depicts 1918 valid states. The path generation evaluated 9068 collision states which are not depicted. The states with collision over the sampled ranges and the non-sampled areas are plotted in white in this section.



(a) (up) Evaluated valid states and optimized interconnected path in the RMP^2 for avoiding a collision using the rotation around the welding direction vector $\delta\alpha$. (down) Robot joint movements through the computed optimized path after mapping the solution found in RMP^2 .



(b) Sequences of robot movements in the simulated robot scene demonstrating the optimized collision avoidance.

Figure 4.4: Collision avoidance in RMP^2 , joint movements, and robot sequences using the process degree of freedom $\delta\alpha$ for avoiding automatically a collision.

For testing the re-configurability of the path generation with respect to changes of the welding process model, a second configuration is evaluated. The DoF of the process is changed to the welding rotation axis $\delta\gamma$. This rotation has less influence on the welding process quality and can be easily prioritized by an expert welder without having experience with robotics, which is one of the main advantages of the proposed approach. For re-configuring this feature, changes of lines 6 and 8 of Listing 4.1 are performed as specified in Listing 4.2. A simulation for two different amounts of times and therefore valid samples (2502 and 225) results in optimized paths as depicted in Figure 4.5. The path does not vary significantly between both plannings. However, the path is smoother when more sampling time is allowed. It is worth noting that if a path is not possible, the path generation returns a message to the end-user stating that no interconnection between the start and goal states was possible.

```

6 <GeoParameter name="α" unit="Deg" type="Fixed" value="45"/>
7 <GeoParameter name="β" unit="Deg" type="Fixed" value="0"/>
8 <GeoParameter name="γ" unit="Deg" type="Range" min="-150" max="50" value="0"/>
9 </FrameConstraintsDoF>

```

Listing 4.2: Semantic changes for adapting the process DoF.

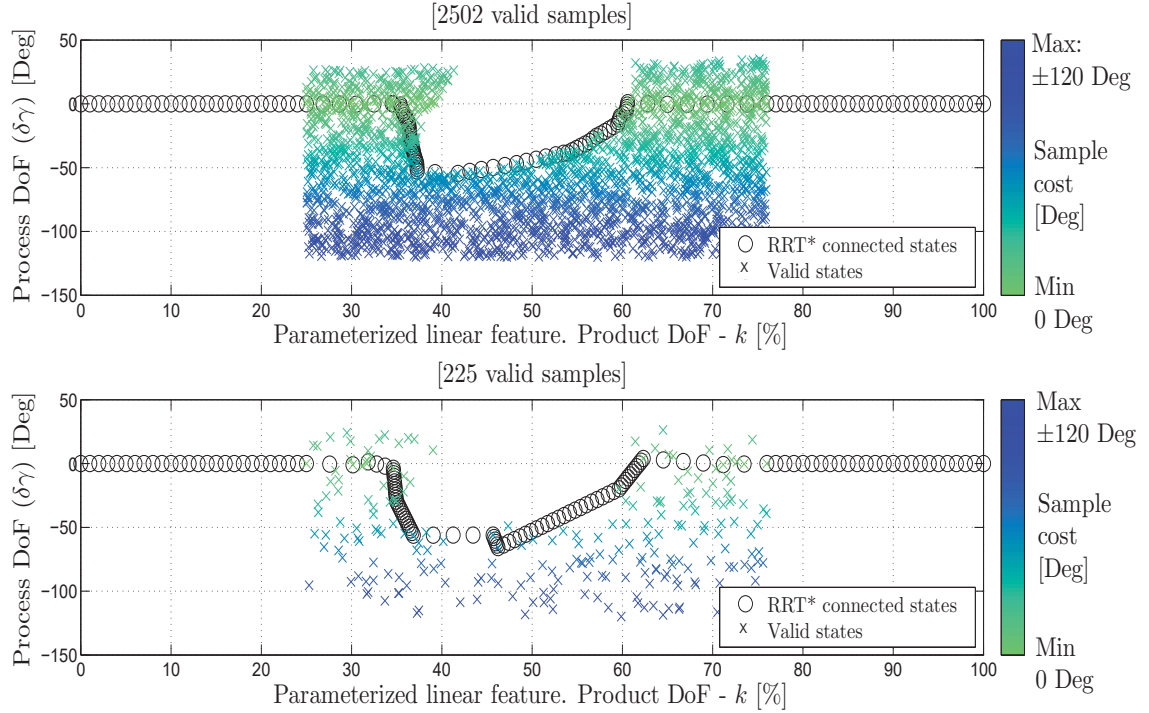
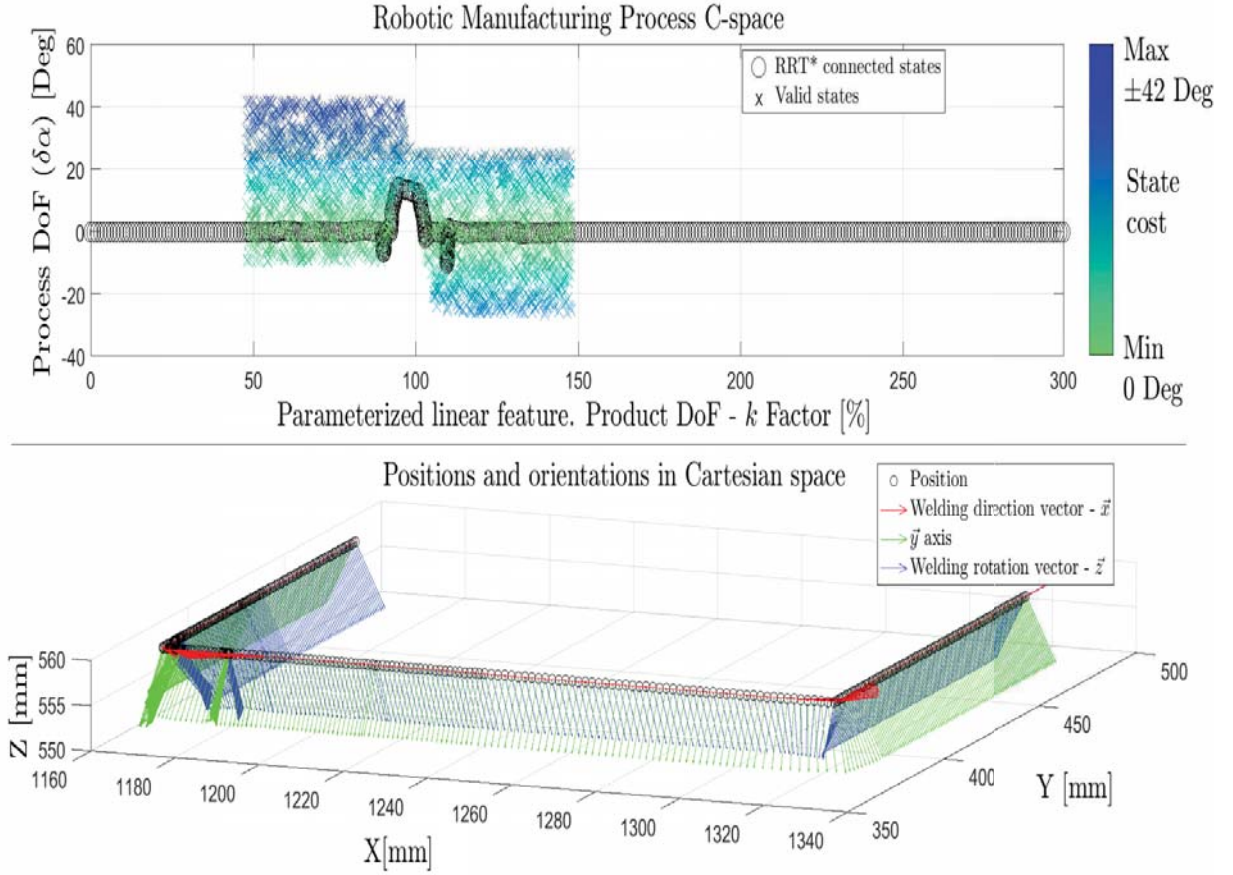


Figure 4.5: Evaluated valid states and optimized interconnected path in the RMP^2 for avoiding a collision using the rotation around the welding rotation vector $\delta\gamma$ for (up) 2502 and (down) 225 valid samples.

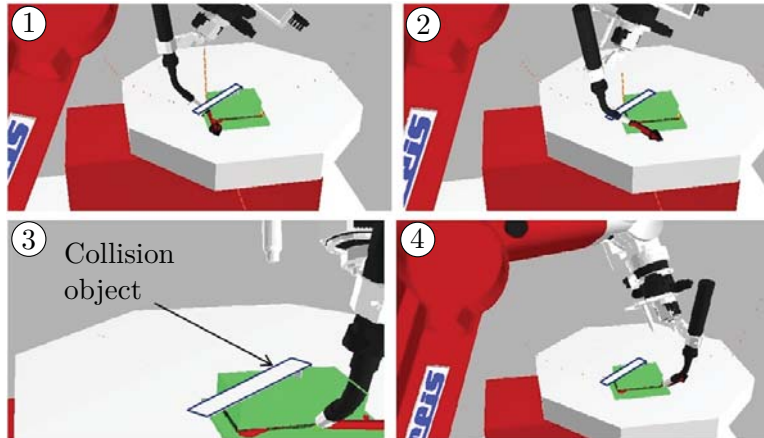
The third configuration evaluates the optimal path generation by simulating a collision avoidance in the corner between the first and second seam using $\delta\alpha$. Three continuous seams were selected in the workpiece depicted in Figure 4.6b, therefore the k dimension is set from 0 to 300. The collision object is also depicted in this figure. The welding direction vector is defined as the DoF of the process $\delta\alpha$.

Figure 4.6 depicts the results in RMP^2 and in the simulation environment. The position and orientation of the robot are also depicted in SE^3 showing the changes of the manufacturing process coordinate frame when avoiding the collision and the optimal rotation from edge to edge as defined in

the product model (corner in the range between 1160 and 1200 [mm] in x). Sequences of the robot simulation are also depicted demonstrating the robot movements achieved by interpreting the PPR model for optimal path generation. The range sampled over the $\mathbb{RMP}^2$ is between 48 [%] and 148 [%] due to the determination of the start and goal states using the function CollisionRange($\cdot$).



(a) (up) Evaluated valid states and optimized interconnected path in the $\mathbb{RMP}^2$ for avoiding a collision using the rotation around the welding direction vector $\delta\alpha$ for multiple seams. (down) Plot of frames in the Cartesian space with the found optimized welding path.



(b) Sequences of Robot movements in the simulated robot scene for the optimized collision avoidance.

Figure 4.6: Optimized robot collision avoidance over two continuous linear features.

Workpiece with semi-circumferential and linear features

In order to evaluate and demonstrate the modularity of the approach, a workpiece with continuous features, first a semi-circumferential and second a linear feature, is tested. Figure 4.7 depicts the workpiece and the collision objects (bars avoiding the optimal welding orientation). The feasibility checking of the optimized path (refer to item 4 in Section 3.4) has been deactivated for demonstrating that the approach also works properly without this add-on (CollisionRange($\cdot$) function not active). The feasible region is defined with the collision detection (refer to Subsection 3.2.1).

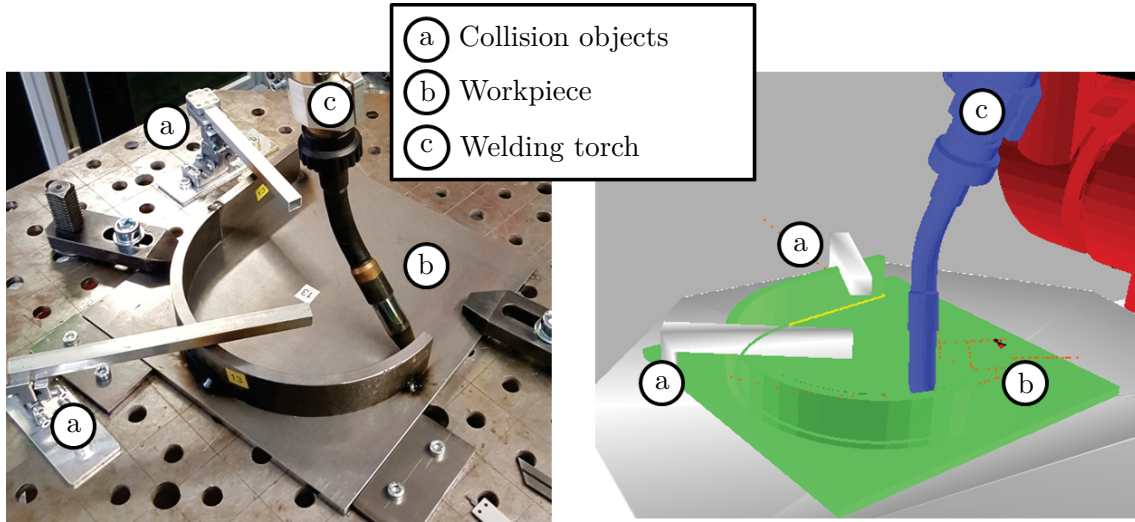


Figure 4.7: Real (left) and simulated (right) workpiece with a semi-circumferential and a linear feature and the collision objects.

The end-user is instructed to select the edges by clicking on them on the desired sequence in the simulation environment. This constitutes, together with the setting of the process model, the task definition. The process model specified in XML is defined for this task with a stickout of 15 mm and also with a fixed rotation of 10 Deg in γ . This angle is configured taking into account the recommendation of experienced welders. Moreover, the maximum and minimum of the DoF $\delta\alpha$ are defined as +45 and -45 Deg respectively. The previous mentioned parameters constitute the process model input for computing the transformation from product to process T_p^{proc} (refer to Section 3.1). Listing 4.3 depicts the changes of the rotational components in the XML. The convention used in the planning is the Euler z,y,x. Figure 4.8 depicts the transformation between the defined product and process coordinate system.

```

6 <GeoParameter name=" $\alpha$ " unit="degree" type="Range" min="-45" max="45" value="45"/>
7 <GeoParameter name=" $\beta$ " unit="degree" type="Fixed" value="0"/>
8 <GeoParameter name=" $\gamma$ " unit="Deg" type="Fixed" value="10"/>

```

Listing 4.3: Process model for the workpiece with semi-circumferential and linear features.

The multiple problem strategy has been configured for this use-case taking into account that the product features are continuous. The product equations are then automatically generated as in Equation (3.5) in page 55, considering the semi-circumferential and linear product models (refer to

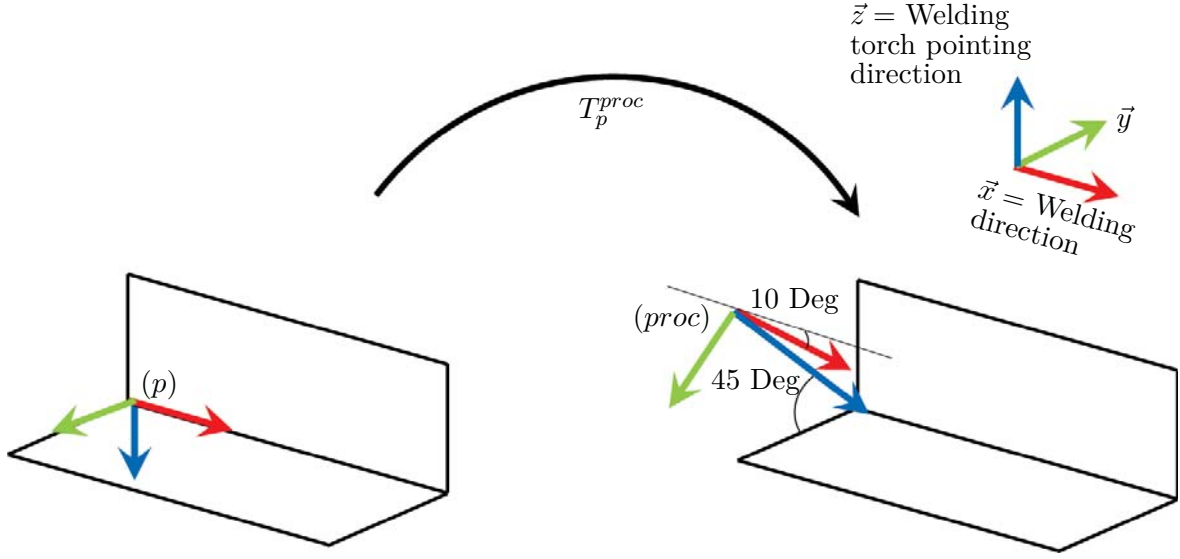


Figure 4.8: Transformation from the product (p) to the process ($proc$) coordinate system for the welding use-case.

Equations (3.2) and (3.1) respectively in page 53), and using the architecture and software implementation presented in Section 3.6. Figure 4.9 shows the state cost elevation map for this path planning problem. In this figure, the collisions between objects in the scene are the non-displayed state costs (transparent spaces).

The termination algorithm, which determines when the sample-based algorithm is coming close to its optimum based on the behavior of the asymptotic optimality, was set in this simulation (refer to item 10 in Section 3.4). The desired delta of motion cost has been set to $\Delta C_{des} = -0.5$. The desire delta of the derivative rule has been set to $\Delta D_{des} = -25$. The minimum number of iterations for the termination algorithm was set to 30 and the number of required repetitions as $r = 3$. Taking into account these conditions, the path planning was performed. The automatic interpretation and generation of equations took 4.26 Seconds (s). The plotting of the RMP C-space, just required for depicting purposes, took 0.759 s. The results and characteristics of the sampling and optimization are summarized in Table 4.1. The heuristic iterations are the ones required by the termination algorithm. The final transformations from RMP^2 to SE^3 and the process finalization took 3.71 s.

Table 4.1: Results and characteristics of the path planning for the semi-circumferential and linear features for the optimized collision avoidance in the robotic welding use-case.

Product Feature	Initial total motion cost [Deg · %]	Final total motion cost [Deg · %]	Evaluated states Nr.	Rewires Nr.	Heuristic iterations Nr.	Duration for optimization and sampling [s]
Semi-circumferential	951.09	506.702	6173	217058	49	30.531
Linear	1731.10	704.603	3109	99747	31	15.387

Figure 4.10 presents the cost evaluations and the fitted function for the automatic termination of the RRT* algorithm. Figures 4.11a and 4.12a depict the motion cost derivative w.r.t. the time for the semi-circumferential and linear features, respectively. Figures 4.11b and 4.12b depict the delta of motion cost for the semi-circumferential and linear features respectively.

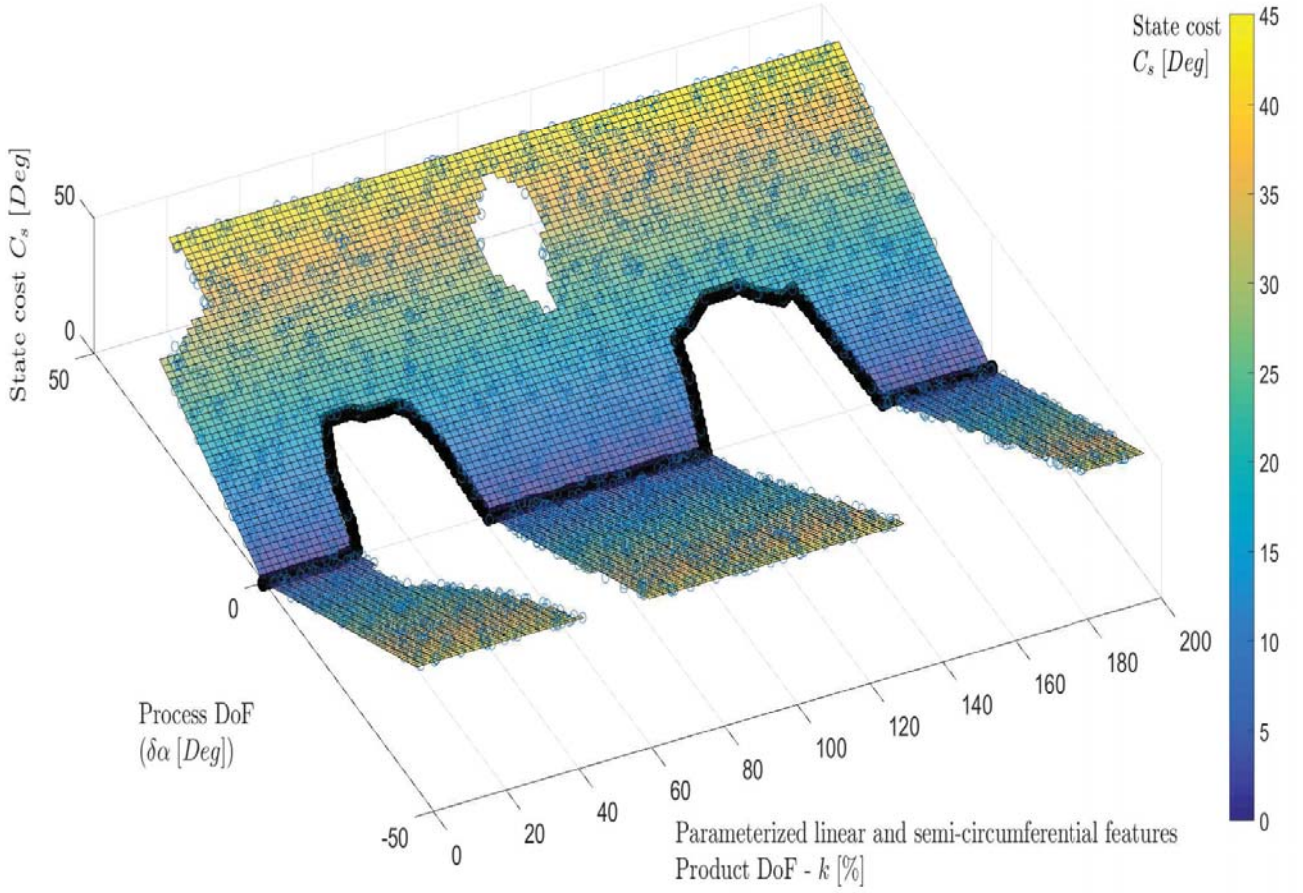


Figure 4.9: State cost elevation map, RMP samples (blue bubbles) and computed path (black bubbles and lines) for the semi-circumferential $k \in [0:100]$ and linear $k \in [100:200]$ product features.

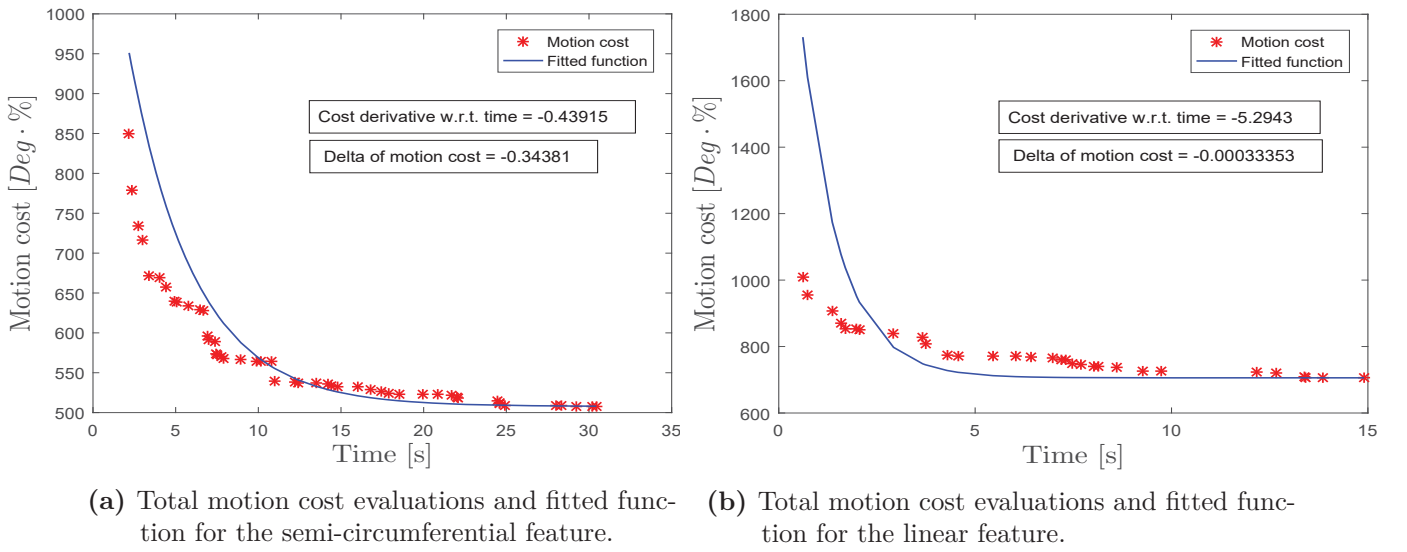


Figure 4.10: Total motion cost evaluations and fitted function for the automatic termination of the RRT*.

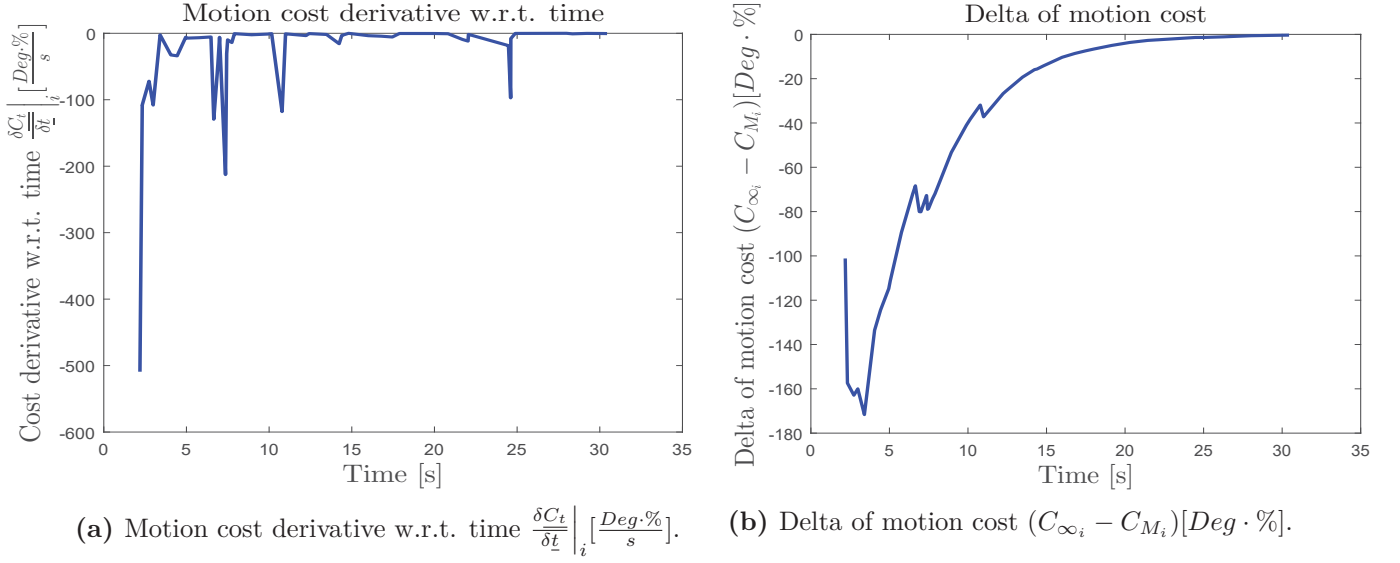


Figure 4.11: Heuristic evaluations for the automatic termination of the RRT* algorithm for the semi-circumferential feature.

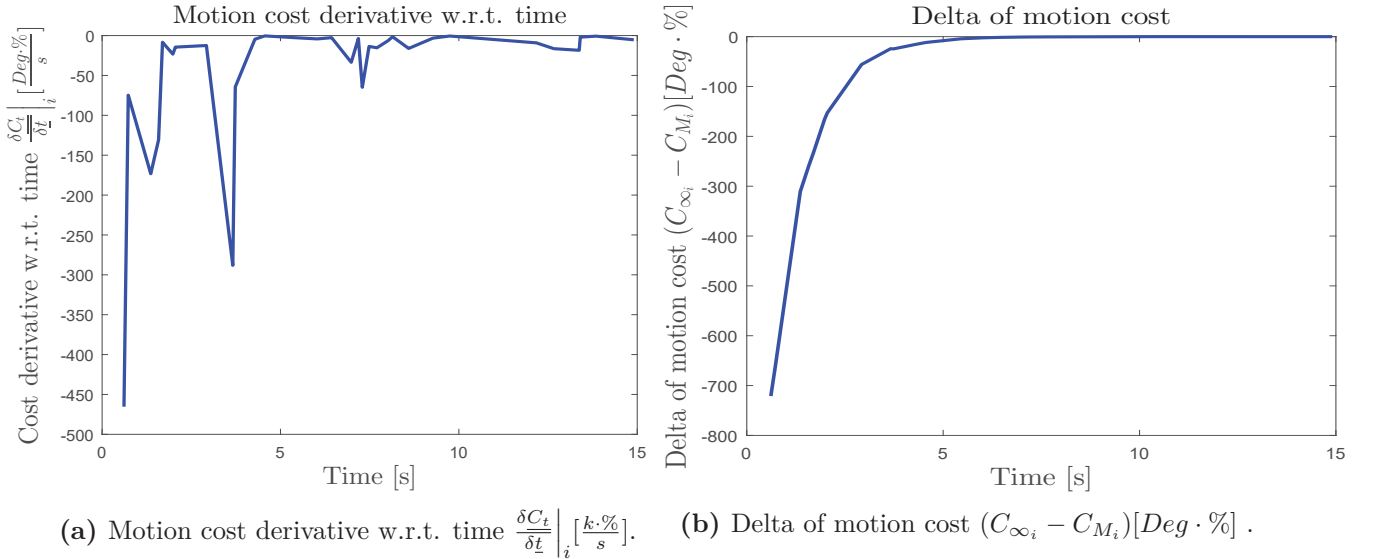


Figure 4.12: Heuristic evaluations for the automatic termination of the RRT* algorithm for the linear feature.

Constraining of the angular velocity through motion cost optimization

Constraining of the angular velocity w.r.t. the k dimension has been simulated for the welding use-case and the collision avoidance over the semi-circumferential feature depicted in Figure 4.7. The angular velocity motion cost, described in Subsection 3.1.2.2, is used to also constrain the rate of change of $\delta\alpha$. The weighting factor w_M has been set to 0.5. In this way, the manufacturing motion cost and the angular velocity motion cost weight 50 percent each. The backward motion policy cost is activated because these movements are undesired for this manufacturing process. The parametrization and equations defined in Subsection 3.1.2.2 in the general definition of the angular velocity motion cost are also valid for this analysis.

For the welding process, the maximum admissible angular velocity has been set as follows ($r_{max} = 5 [\frac{Deg}{\%}]$). Path planning simulations for desired angular velocities r_d equal to 0.1, 1, 2, and 4 $[\frac{Deg}{\%}]$ (ratio border value b_v equal to 0.02, 0.2, 0.4, and 0.8, respectively) are plotted in Figure 4.13. The collisions between objects in the scene are the non-displayed state costs (transparent spaces) over the $\mathbb{RMP}^2$ s. The planning was allowed for 240 s using an automatically set RRT* algorithm, which was considered a proper time to optimize the total motion cost.

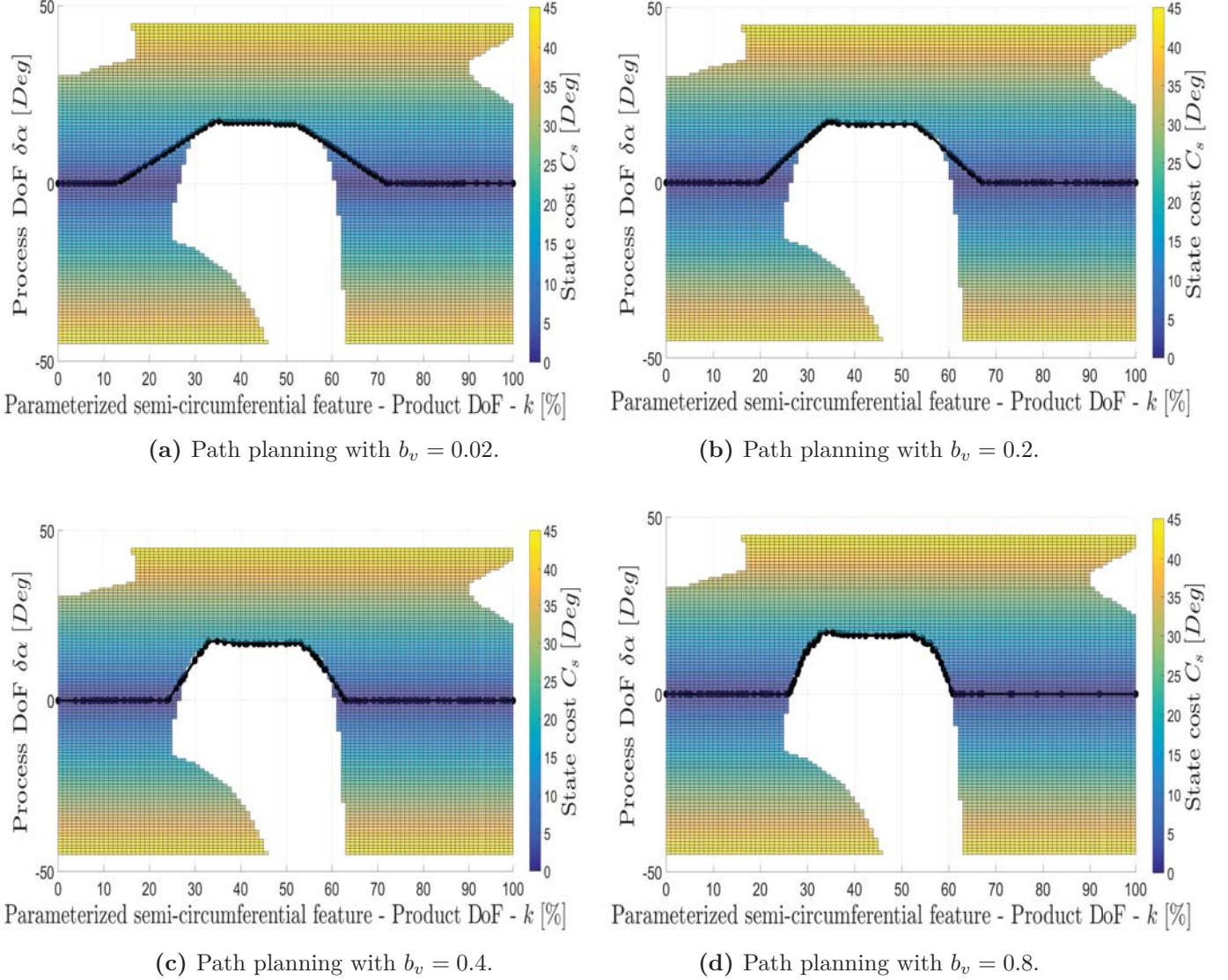


Figure 4.13: Path planning (black bubbles and lines) over the welding use-case $\mathbb{RMP}^2$ constraining the angular velocity w.r.t. k with different b_v factors for the collision avoidance through the semi-cumferential feature.

Table 4.2 summarizes the results and characteristics of the plannings. The "initial and final total motion costs" values from Table 4.2 show the costs when constraining the angular velocity. Lower values of the "ratio border value" represent greater "final total motion costs" due to the restriction to fast angular rates. This restriction does not allow the approach to plan paths with states which have better state costs but which represent a faster angular rate as in the different paths of Figure 4.13. Note that the shown results of the constraining of the angular velocity also takes into account the manufacturing cost and therefore the relation between total costs and ratio border values is not linear.

Table 4.2: Results and characteristics of the path planning constraining the angular velocity w.r.t. k for different ratio border values in the welding use-case.

Ratio border value	Initial total motion cost [Deg · %]	Final total motion cost [Deg · %]	Evaluated states Nr.	Rewires Nr.
0.02	3254.57	1274.589	73607	3406672
0.2	1366.89	736.804	76841	3571606
0.4	1532.43	599.957	85386	4007401
0.8	1326.21	538.542	75310	3493525

Path planning with more than one process DoF

Planning with more than one process DoF is also possible with the proposed approach and it has been simulated for the welding use-case. In order to depict this, the planning was performed using $\delta\alpha$ and $\delta\beta$ as DoFs with the bounds specified in Listing 4.4. The path planning was allowed for a maximum time of 1200 s and performed for the semi-circumferential feature. As in the previous plannings, the RRT* was used. The termination condition algorithm was active in order to demonstrate that this algorithm is independent of the problem dimensionality.

```

<GeoParameter name="α" unit="degree" type="Range" min="-45" max="45" value="45"/>
<GeoParameter name="β" unit="degree" type="Range" min="-90" max="90" value="0"/>
<GeoParameter name="γ" unit="degree" type="Range" min="0" max="0" value="0"/>

```

Listing 4.4: Semantic modeling of the welding process in XML format for the collision avoidance using $\delta\alpha$ and $\delta\beta$ as DoFs specifying its bounds.

Figure 4.14 depicts the path planning (black bubbles) and the RMP³ for the optimized collision avoidance using both DoFs. Figure 4.15 complements the previous plot by showing two perspective views of this RMP³. The state costs in the RMP C-space has been plotted translucent in order to observe the costmap. Collisions are plotted as null values in the previously introduced plots (less colored areas). The planning optimizes the path in order to keep the optimal cost states. In order to depict the collision avoidance, Figure 4.16 depicts two perspective views of the path and the collision states demonstrating the functioning of the algorithm. Table 4.3 summarizes the required times for the planning and Table 4.4 the results and characteristics of the sampling and optimization.

Table 4.3: Required path planning times in the robotic welding use-case using $\delta\alpha$ and $\delta\beta$ as process DoFs.

Interpretation [s]	Plotting of space [s]	Sampling and optimization [s]	Transformations [s]	Total [s]
3.74	109.37	303.96	3.75	420.82

Table 4.4: Results and characteristics of the sampling and optimization in the robotic welding use-case using $\delta\alpha$ and $\delta\beta$ as process DoFs.

Initial total motion cost [Deg · %]	Final total motion cost [Deg · %]	Evaluated states Nr.	Rewires Nr.	Heuristic iterations Nr.	Duration for sampling and optimization [s]
2116.94	471.77	20980	759383	35	303.96

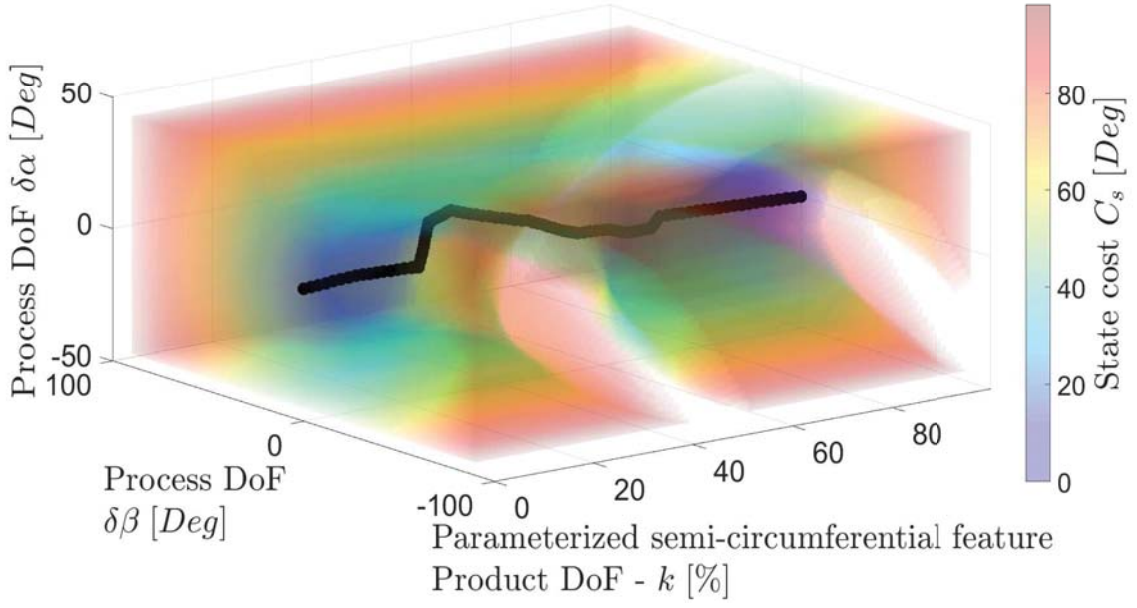


Figure 4.14: 3D representation of the planned path (black bubbles) in the $\mathbb{RMP}^3$ for the optimized collision avoidance of a semi-circumferential feature using $\delta\alpha$ and $\delta\beta$ as process DoFs.

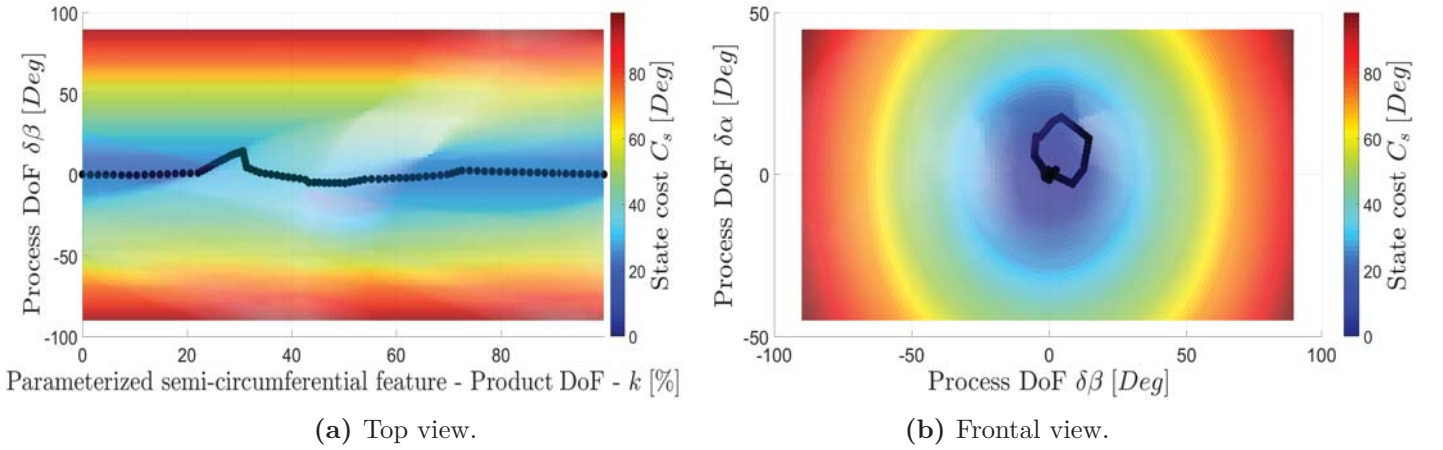


Figure 4.15: 2D-projections of the planned path in the $\mathbb{RMP}^3$ depicted in Figure 4.14.

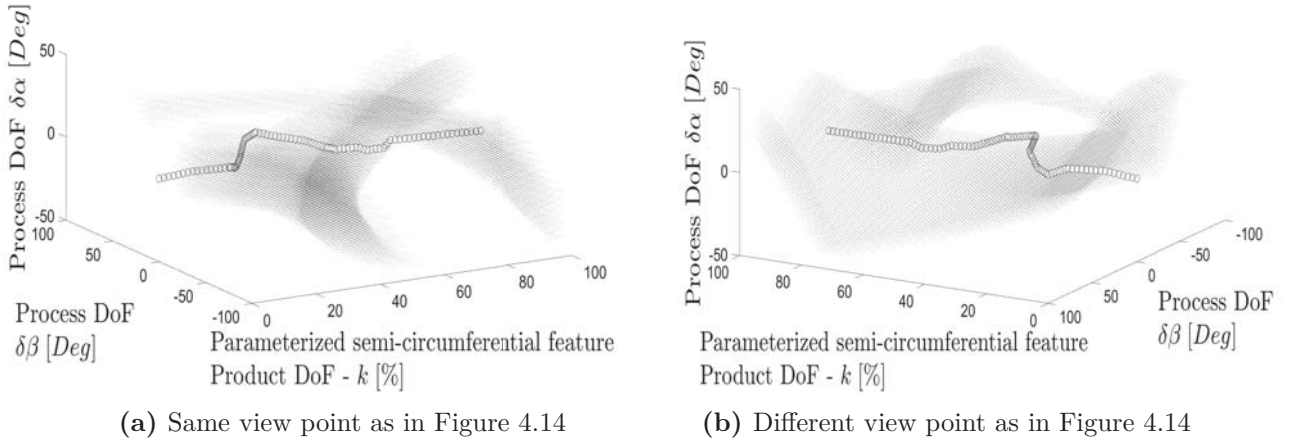


Figure 4.16: 3D views depicting the collision (points) and the planned path (bubbles) of the representation depicted in Figure 4.14.

Planning using other algorithms

Previously presented plannings were all performed using the RRT* algorithm. A benchmark among different optimal sample-based planners is offered for the collision avoidance of the semi-circumferential part. These algorithms are the RRT*, the informed RRT*, the T-RRT, and the BIT*. Each algorithm was run for 50 times after the interpretation of the manufacturing task and its completion time and final total cost was recorded. The manufacturing task was set as the planning of the semi-circumferential path using $\delta\alpha$ as DoF.

The automatic termination heuristic (refer to step 10 in Section 3.4) was set for the different algorithms except for the T-RRT. The latter algorithm terminates automatically when the tree reaches a solution. The optimization of this algorithm is performed over each tree extension. The desired delta of motion cost for the termination algorithm has been set to $\Delta C_{des} = -0.5$. The desired delta of the derivative rule has been set to $\Delta D_{des} = -25$. The minimum number of iterations for the termination algorithm was set to 30 and the number of required repetitions to $r = 3$. The maximum planning time (t_{max}) was set to 90 s. The state and total motion cost defined in Section 4.3 was set for all the algorithms.

Boxplots for the obtained total motion costs and the required planning durations are depicted in Figure 4.17 and Figure 4.18, respectively. The boxes represent the 25th and 75th percentile (p_1 and p_3 respectively). The whiskers ($p_3 + 1.5 \cdot (p_3 - p_1)$ and $p_1 - 1.5 \cdot (p_3 - p_1)$) extend to the most extreme data points not considered outliers. The outliers are plotted individually using the + symbol. No specific configurations were applied to the algorithms with respect to the OMPL default configurations. The only algorithm which required settings was the BIT*. The PruneThresholdFraction parameter was set to 0.2, the RewireFactor parameter to 0.2 and the SamplesPerBatch to 250. The modified parameters are defined in the implementation of this algorithm in the OMPL source code.

The RRT* presents the best cost-time performance as seen in both figures. The final total motion cost converge more to the optimum for the RRT* and the informed RRT* algorithms and their variability is less in comparison to the other two algorithms as shown in Figure 4.17. The algorithm with the highest final total motion cost is the T-RRT. The median, percentiles, and whiskers of the T-RRT are higher in comparison to the other algorithms. Moreover, this is the algorithm with more outliers, namely four. This is associated to the exploration nature of the T-RRT. This algorithm optimizes on the search through the goal and no further optimization is done after connecting a path.

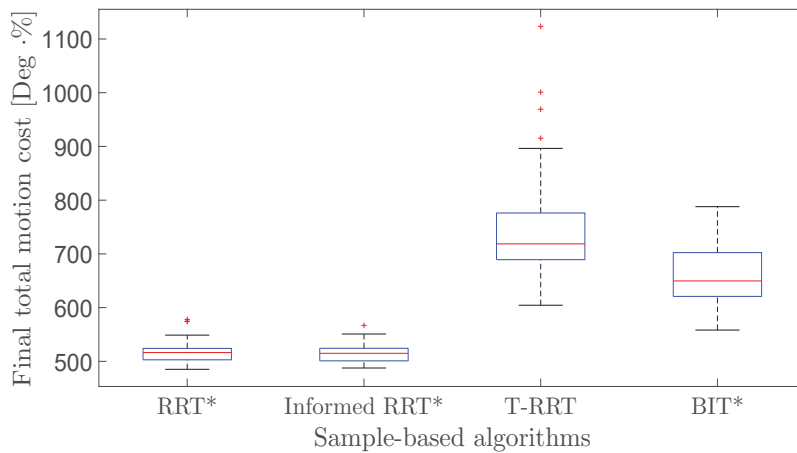


Figure 4.17: Comparison of the final total motion cost with different optimal sample-based algorithms for the collision avoidance of the semi-circumferential feature.

Results do not vary for the planning duration as drastically as for the final total motion costs among the sample-based algorithms except for the BIT*. The BIT* never triggered the automatic termination heuristic in the 50 iterations. Instead the maximum planning time always triggered the termination condition as it can be seen in Figure 4.18. The fastest triggering of the automatic termination was achieved with the RRT* with 8.38 s. This algorithm presents also the fastest median 19.99 s. The less variability of planning durations is observed for the T-RRT. The RRT* and the informed RRT* present outliers just when the termination condition is not triggered. This benchmark demonstrates that the proposed approach allows the use of different sample-based algorithms for the optimization of manufacturing processes. Figure 4.19 depicts the benchmarked algorithms showing the vertices of the planning.

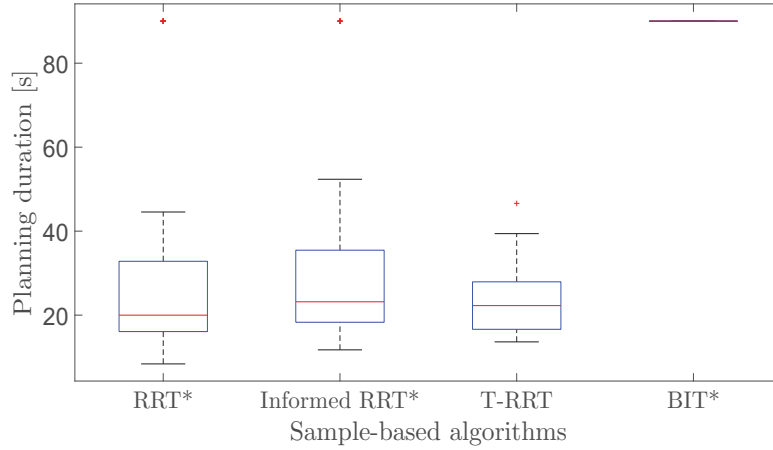


Figure 4.18: Comparison of the required planning durations with optimal sample-based algorithms for the collision avoidance of the semi-circumferential feature.

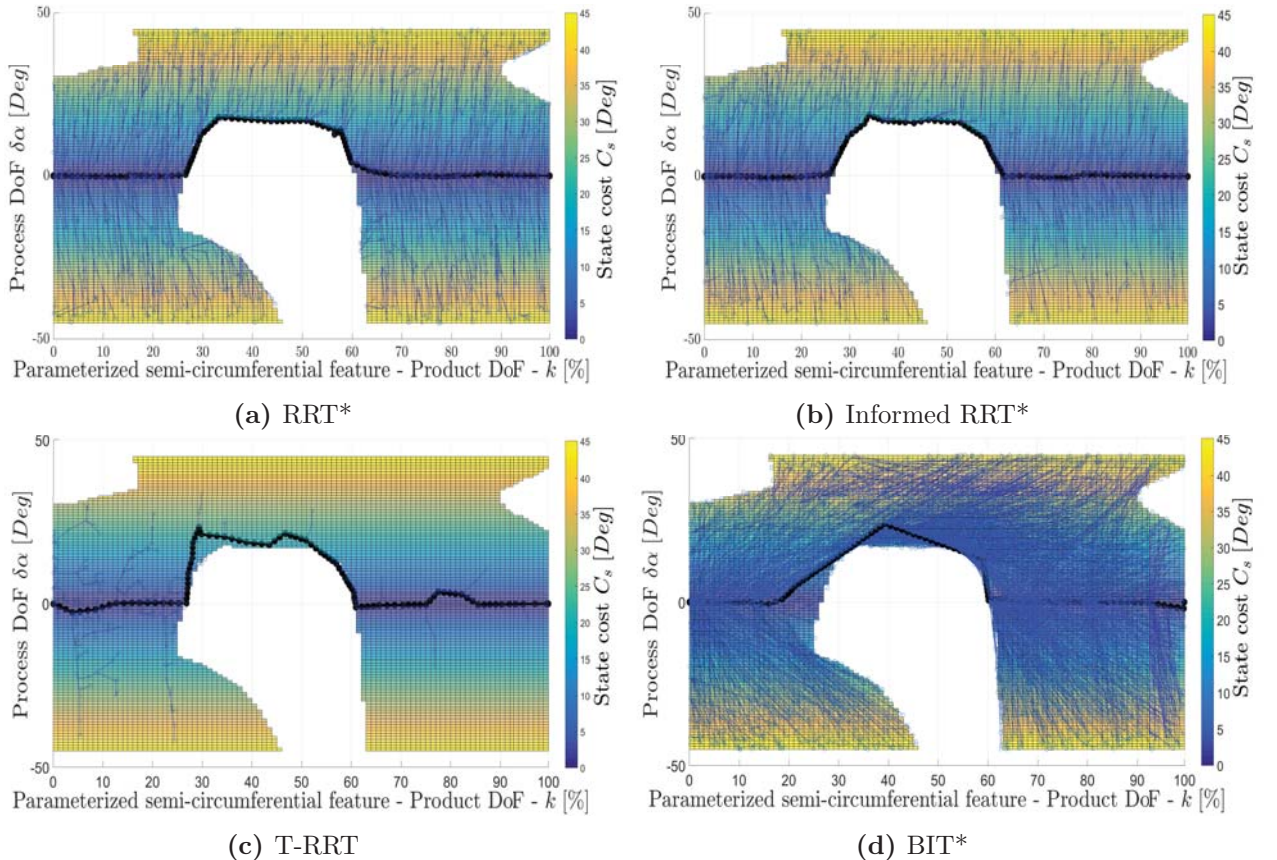


Figure 4.19: Different path planning algorithms depicting their vertices and computed paths.

4.5.3 Welding Robot Experimentation

Two experiments have been approached for the evaluation of this use-case. The first experimentation has been performed for programming the industrial robot described in Subsection 4.5.1 with the proposed AOLP and with the teach-in approaches, using the workpiece with semi-circumferential and linear features (refer page 91 and Figure 4.7) while having to avoid collisions which do not allow keeping the most optimal orientation. The required programming time and final process quality was evaluated in this experiment. Nine participants performed the programming with the mentioned approaches.

Furthermore as second experiment, a comparison of the required programming times using the AOLP and a commercial OLP software was also performed. In this experiment, six participants without previous experience with the internal developed software and the commercial robot programming tool Delfoi simulated the task. Focus of this experimentation is the required programming time. Initially, the coordinate systems defined in the product and in the robot welding tool were explained to the participants. Figure 4.8 in page 92 depicts the coordinate systems introduced to the participants.

Experiment 1: Comparison between the proposed AOLP and the teach-in approaches

Experimental task description

The task description given to the nine participants (two women and seven men) was as follows: The same workpiece type, continuous semi-circumference (first) and linear feature (second) must be welded with the following considerations: First, the welding torch should have a distance of 15 mm with respect to the welding seam for assuring a proper stick-out for the process. Second, in order to get a good welding quality, 45 Deg should be kept as possible between the two welding-seam surfaces. Third, the $\vec{x}$ vector (welding direction) should have 10 Deg difference with respect to the welding seam in order to assure good welding quality. Fourth, the only DoF of the tooltip is the rotation around the $\vec{x}$ vector. The maximum and minimum deviations from the desired configuration are +45 Deg and -45 Deg in the DoF $\delta\alpha$.

The participants were asked to perform the task in the fastest and most optimal way. The optimality of the task was explained as trying to keep the most optimal welding orientation and distance in respect to the welding seam while avoiding the collisions as specified in the task description. The participants were provided with a vernier caliber and a half circle protractor for measuring, if desired, the TCP distance and orientation in respect to the welding seam while manually teaching the welding task.

Experimental procedure

Documentation of the programming with both approaches was provided to the participants. Explanation of each approach before performing the task was provided and the time the process took was measured. For the programming via AOLP, the participants were asked to configure the process model and to generate a program automatically with the simulation environment. Required times for both tasks were also measured. For the teach-in approach, the teaching time was measured up to completion before testing the program. The localization of the workpiece was done previously using a laser-tracker AT-901 for the AOLP approach taking into account that this is not part of the automation implemented during this work. In order to compensate this, all participants received an initial program for the teach-in programming where the approach and retrieve paths were already established. The

initial program defined the start and the end robot pose for the welding. Additionally, the welding control initialization was linked as required to the robot program.

Experimental results

Comparison between the required programming time and the welding quality is the focus of this experimentation. Each participant generated programs with both approaches. Figure 4.20 depicts the sequence of an automatically generated robot program for the welding process avoiding optimally collisions while satisfying the continuous semi-circumferential and linear features' constraints in the simulation environment and in the robot cell. The RMP² and one path planning as example for this experimentation has been depicted in Figure 4.9 in page 93.

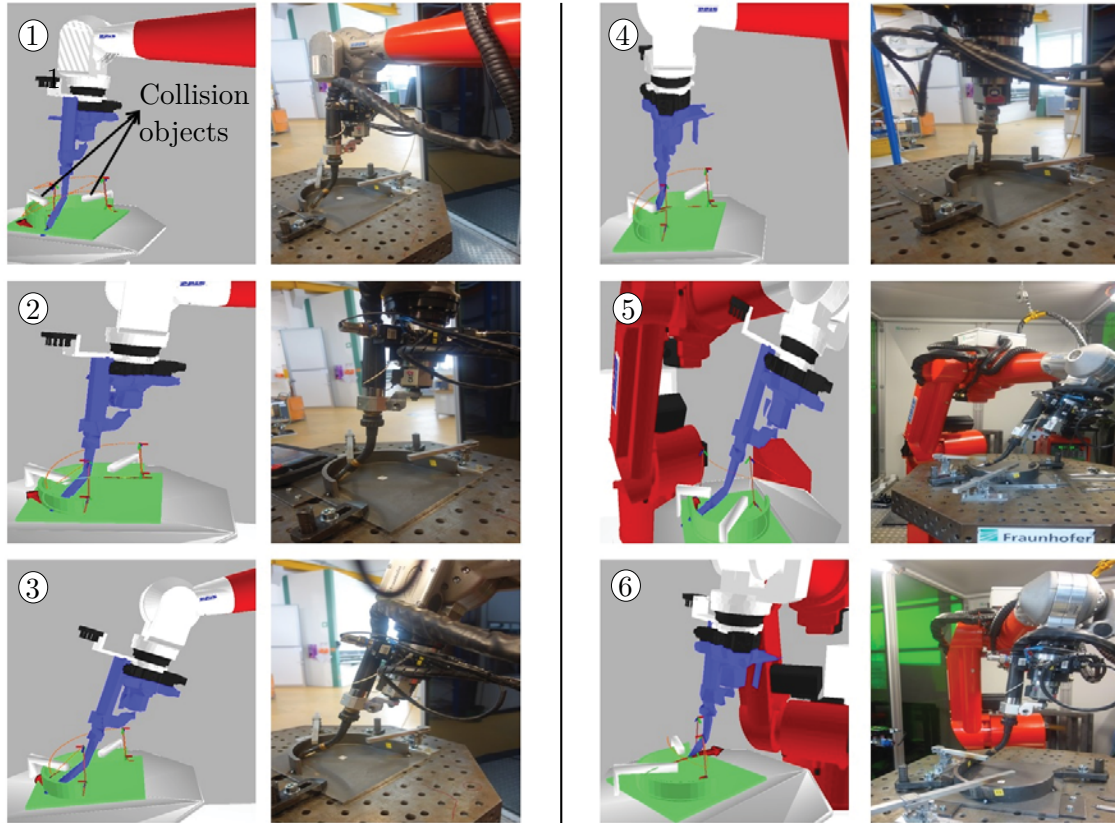


Figure 4.20: Sequence of robot movements in simulation and with the real robot for the optimized collision avoidance for welding a workpiece with continuous semi-circumferential and linear features.

Figure 4.21 summarizes the required time to program the robot by each participant. From the nine participants, three of them had no previous experience programming robots (participants 2, 4, and 9), three had previous experience programming robots different than Reis (participants 1, 3, and 8), and three of them had experience programming Reis robots (participants 5, 6, and 7) and knew beforehand the AOLP approach. The programming time average for each group of participants illustrating the influence of programming experience is depicted in Figure 4.22. The average programming time of all participants is also depicted in this figure. The measured required times for the explanation, configuration, and planning of the proposed approach are faster than the existing teach-in method for the given collision avoidance task. Factors of the AOLP being faster than the teach-in approach are listed in Table 4.5.

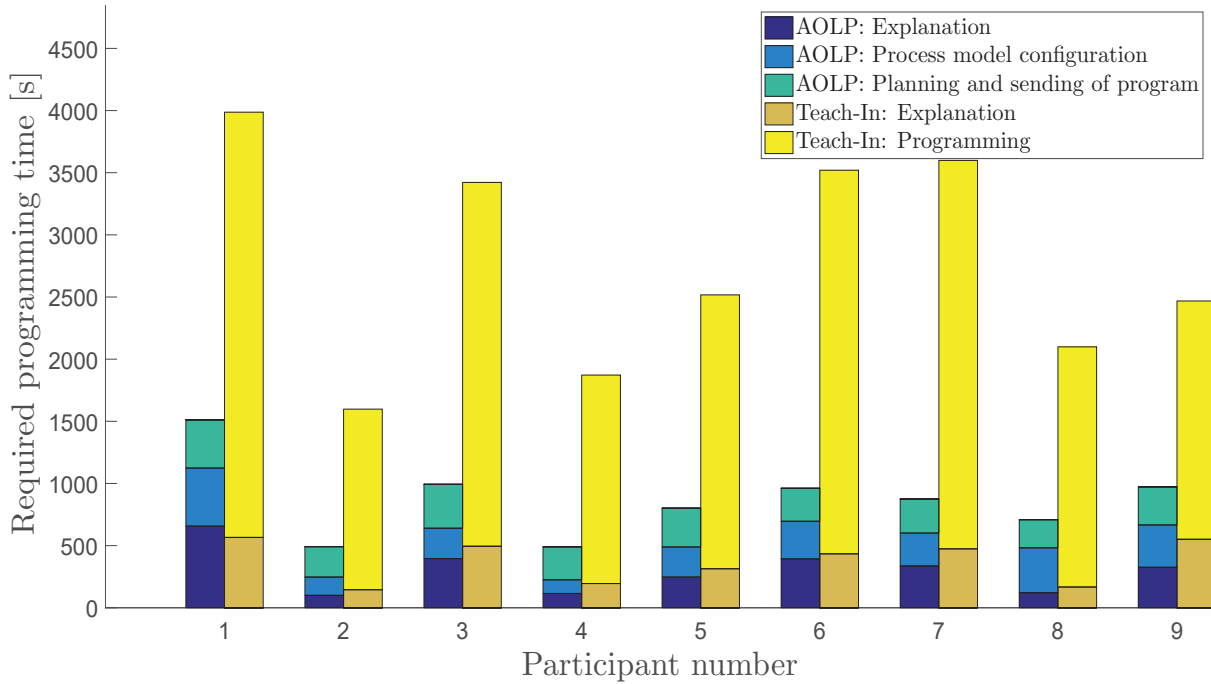


Figure 4.21: Required programming time for the welding task involving collision avoidance using the Automatic Optimized Offline Programming (AOLP) and the teach-in approaches for all participants.

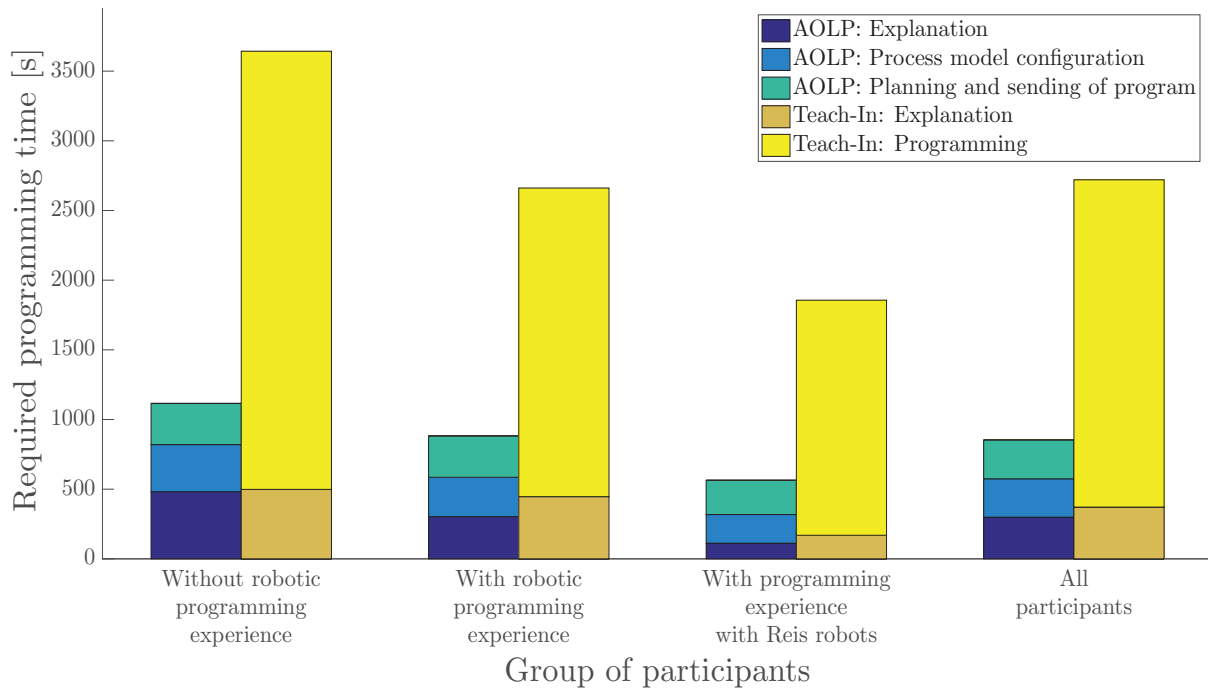


Figure 4.22: Influence of the programming experience on the required programming time for the welding task involving collision avoidance with the AOLP and the teach-in approaches for the groups of participants.

Under the label "AOLP: planning and sending of program", the time required for configuring the simulation environment through the programmed macros, the clicking over the desired weld seams, and

Table 4.5: Factors of the AOLP being faster than the teach-in approach.

Task	without robot programming experience	with robot programming experience	with Reis programming experience	All Participants
Explanation	1.03	1.47	1.50	1.24
Configuration and planning	4.96	3.81	3.73	4.23

the required planning time of the proposed approach was measured. Table 4.6 lists the required times to compute an optimized solution. The time required for the proposed approach is subdivided into the interpretation of the PPR components, the plotting of the space (just required for visualization purposes), the sampling and optimization, and the transformations of the solution from the RMP C-space ($\mathbb{R}^{m \times n}$) to the Cartesian space ($\mathbb{SE}^3$). The slight differences in the interpretation, plotting of space, and transformations times are associated to the CPU and RAM utilization at each planning time. Variations between the sampling and optimization times are due to the automatic termination algorithm (refer item 10 in Section 3.4). Parameterization of this algorithm applies as specified in page 92. After the computation, the participants sent a program automatically to the Reis controller by using a programmed macro. The maximum and minimum computation time taking into account the given conditions were 92.21 s and 39.64 s, respectively.

Table 4.6: Required programming time for the path planning for the semi-circumferential and linear features for all participants.

Task	Participant								
	1	2	3	4	5	6	7	8	9
Interpretation [s]	3.80	4.64	3.72	5.31	3.68	4.82	3.82	4.44	3.67
Plotting of space [s]	0.76	0.76	0.72	0.80	0.75	0.84	0.79	0.75	0.74
Sampling [s]	42.51	54.10	40.63	80.79	32.54	48.15	49.55	30.01	42.91
Transformations [s]	3.80	4.64	3.73	5.31	3.68	4.83	3.82	4.44	3.68
Total [s]	50.87	64.14	48.8	92.21	40.65	58.64	57.98	39.64	51.00

The welding quality was evaluated by measuring twenty welding throat thicknesses for each workpiece. The throat thickness is defined as by the International Organization for Standardization (1992). The throat thickness was measured with a 2D laser scanner, scanControl 2900-50 from the company Micro-Epsilon mounted on the robot flange, as depicted in Figure 4.23, and by using a Fraunhofer IPA internally developed tool for computing it automatically, given as input the profile to be evaluated.

Figure 4.23 depicts the twenty laser profiles taken for measuring the throat thicknesses after overlaying images of each profile for each workpiece. The five first measurements were taken before the first collision avoidance. The five next measurements were taken during the collision avoidance of the semi-circumferential part. The next five after this collision avoidance, and the last ones in the linear feature where the second collision was avoided.

For each workpiece, the mean of the throat thicknesses was calculated and the absolute deviation of each measurement with respect to the workpiece mean was taken into account in order to evaluate the homogeneity of the welding. Figure 4.24 depicts the absolute errors w.r.t. the mean of the throat thicknesses for each participant and method.

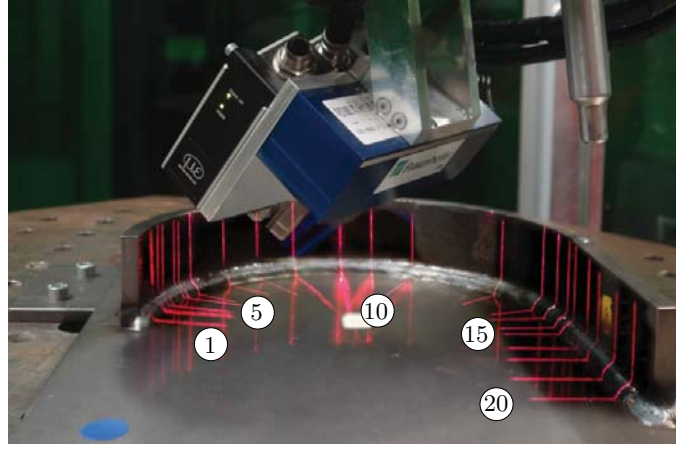


Figure 4.23: Laser scanner with the twenty measured throat thicknesses after overlaying images of each taken profile.

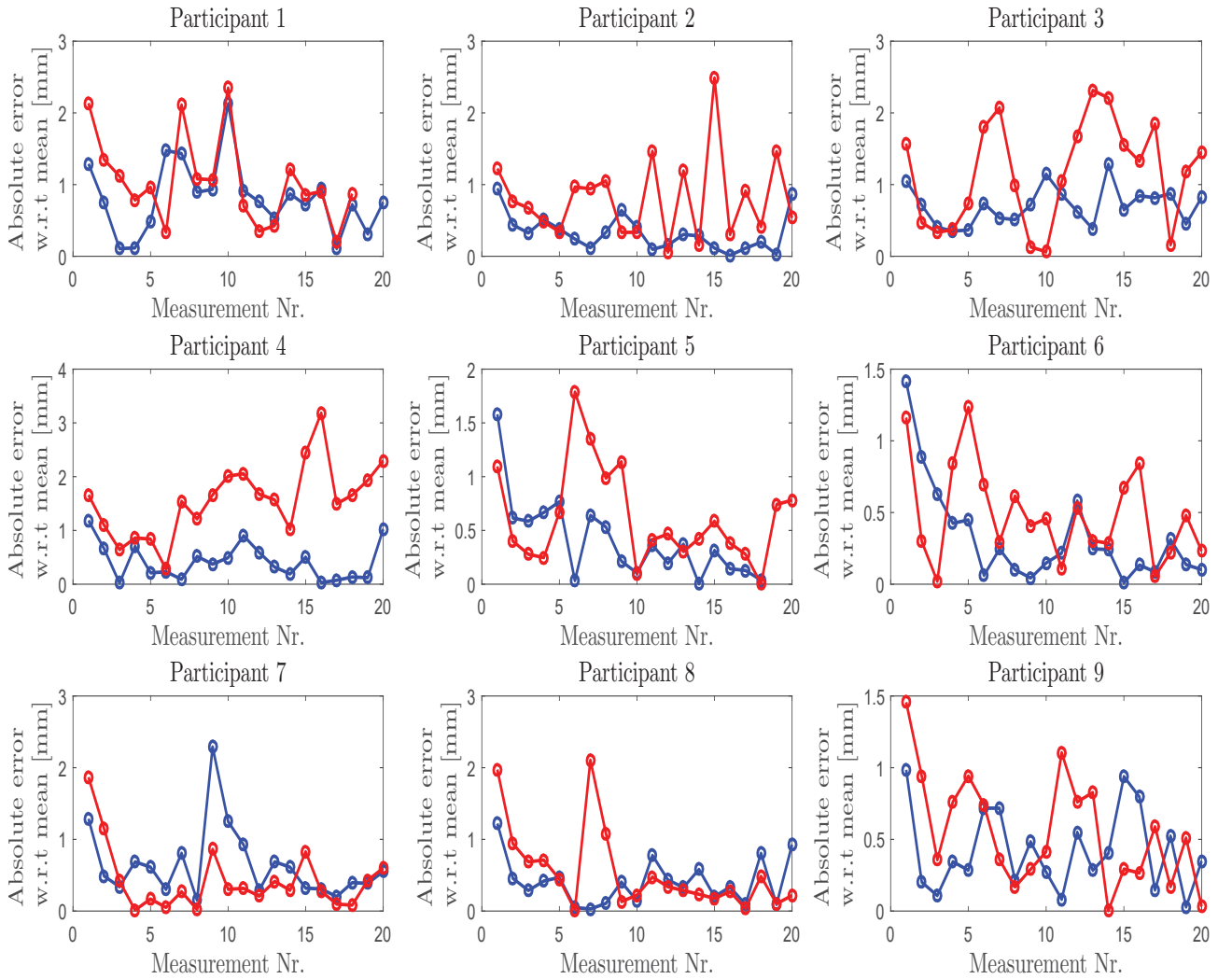


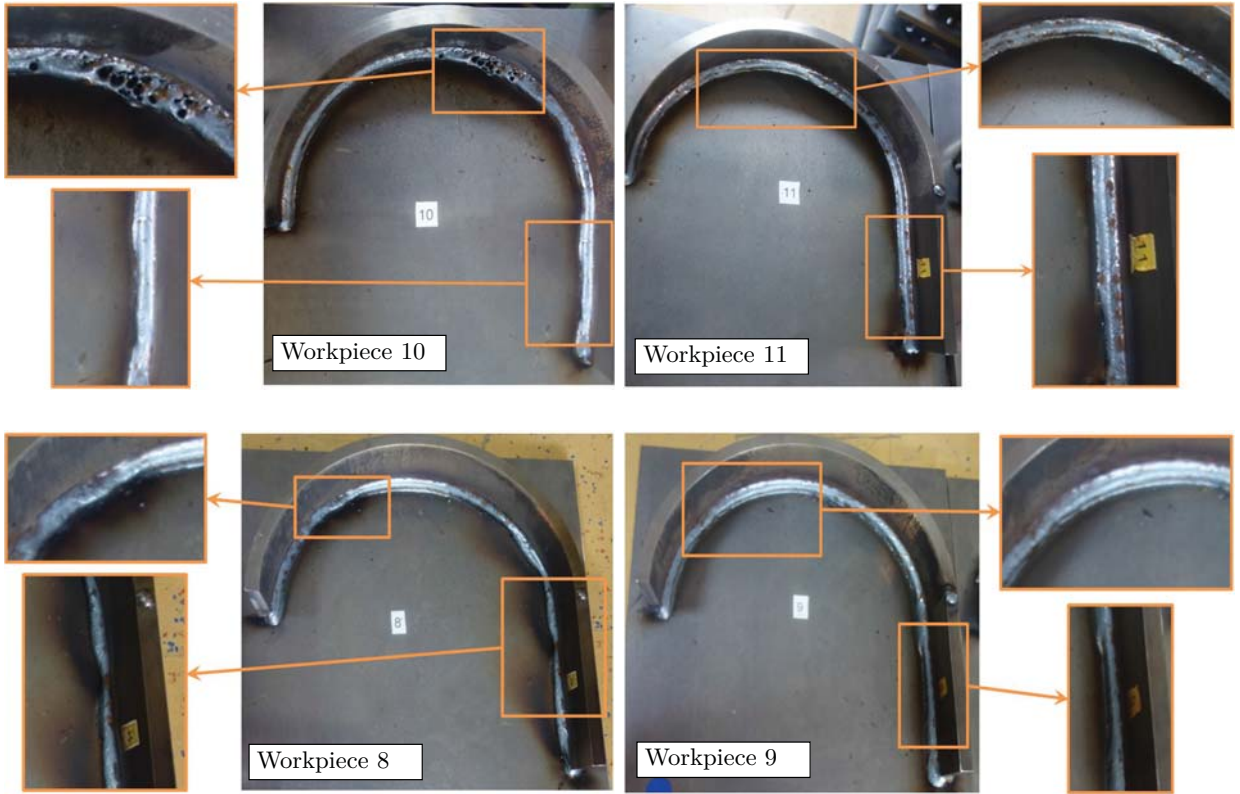
Figure 4.24: Absolute errors w.r.t. the mean of the throat thicknesses. Red points represent the teach-in approach and blue the proposed AOLP. Measurement 1-5: before the first collision, measurement 6-10: avoiding collision of the semi-circumferential feature, measurement 11-15: after first collision avoidance in the circumferential feature, measurement 16-20: in the linear feature.

Table 4.7 lists the mean of the absolute error w.r.t. the mean of the throat thicknesses for each participant and method depicting that the AOLP is more homogeneous than the teach-in. The throat thicknesses homogeneity improved 35.62 %, in comparison with the teach-in, using the proposed AOLP approach for this specific experimentation.

Figure 4.25 depicts some examples of the final quality of the welded workpieces with both methods. Workpieces 10 and 11 were welded by participant 2 and workpieces 8 and 9 by participant 3. Table 4.8 and Table 4.9 summarize the results and characteristics of the path planning for the semi-circumferential and linear features respectively for all the participants. The planning process of the semi-circumferential feature of participant four did not trigger the termination algorithm because several initial total motion costs including backward motions affected the fitting function.

Table 4.7: Mean of the absolute error w.r.t. the mean of the throat thickness for each participant and method.

Method	Participant									All - average
	1	2	3	4	5	6	7	8	9	
AOLP [mm]	0.35	0.19	0.20	0.28	0.28	0.24	0.34	0.24	0.23	0.2611
Teach-In [mm]	0.45	0.44	0.61	0.50	0.35	0.26	0.31	0.42	0.31	0.4056



(a) Example of workpieces welded using the teach-in approach.

(b) Example of workpieces welded using the AOLP approach.

Figure 4.25: Example of welded workpieces using the Automatic Optimized Offline Programming (AOLP) and the teach-in approach.

Table 4.8: Results and characteristics of the path planning for the semi-circumferential feature for the different participants.

Participant	Initial total motion cost [Deg · %]	Final total motion cost [Deg · %]	Evaluated states Nr.	Rewires Nr.	Heuristic iterations Termination alg. Nr.	Duration of optimization [s]
1	1192.92	517.160	4167	139236	48	25.13
2	1212.41	503.297	9255	342127	56	37.57
3	1096.85	530.112	3285	106259	31	20.26
4	1187.63	492.352	19283	776623	Max. Time	60
5	1062.85	532.153	2306	70931	31	17.06
6	915.92	508.023	6622	235018	35	33.84
7	1378.23	541.453	3545	115879	31	21.92
8	1394.28	549.832	2440	75621	35	18.30
9	1272.05	510.819	6124	215098	56	28.50

Table 4.9: Results and characteristics of the path planning for the linear feature for the different participants.

Participant	Initial total motion cost [Deg · %]	Final total motion cost [Deg · %]	Evaluated states Nr.	Rewires Nr.	Heuristic iterations Termination alg. Nr.	Duration of optimization [s]
1	1554.32	704.385	4965	169746	42	17.38
2	1246.07	727.375	5469	189402	44	16.52
3	1424.32	689.084	6972	249018	31	20.37
4	1581.78	712.562	4078	135854	37	20.79
5	1500.88	706.495	4899	167172	36	15.47
6	1265.55	704.277	3875	128140	34	14.31
7	1537.41	686.863	9643	358347	62	27.63
8	1593.62	704.286	3325	107739	37	11.71
9	1246.07	705.417	4522	152726	43	14.40

Experiment 2: Comparison of required programming time with a commercial OLP system

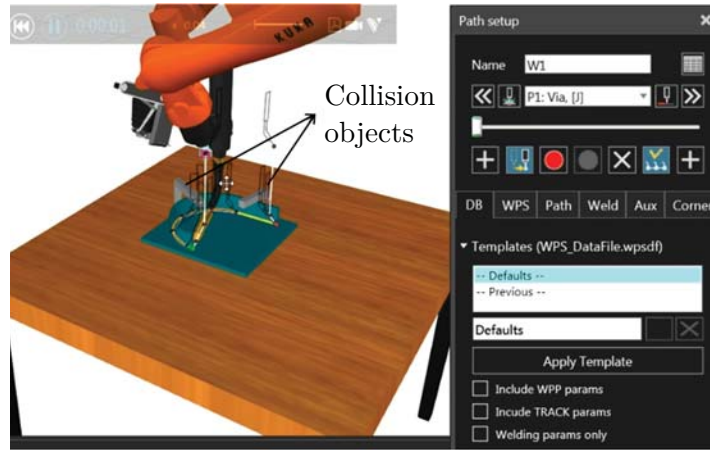
The same task evaluated above (welding of the semi-circumferential and linear features avoiding collisions) was simulated by six participants using the commercial software Delfoi and the proposed AOLP approach using the internal developed simulation. A document describing the task was given to each participant as in the previous experiment. Additionally, a video showing the programming in each simulation environment was prepared and given to the participants. Each participant was allowed to watch each video for 5 minutes. Support was given during the simulation as requested for the usage of the commercial and the internal developed software. Figure 4.26 depicts both simulation environments.

The DelfoiArc toolbox was used for the configuration of the simulation in Delfoi. This toolbox allows the automatic planning based on geometrical features. The geometrical approach allows generating robot poses based on the workpiece geometry without reacting to infeasibilities. In this way, end-users were guided to plan a path for the semi-circumferential and the linear features based on the CAD model. Collisions have to be avoided by adding poses and re-teaching the geometrically generated poses. The internally developed simulation environment included the AOLP approach and used an RRT* as in the previous experiment for optimizing the path. As in the previous experiment, the

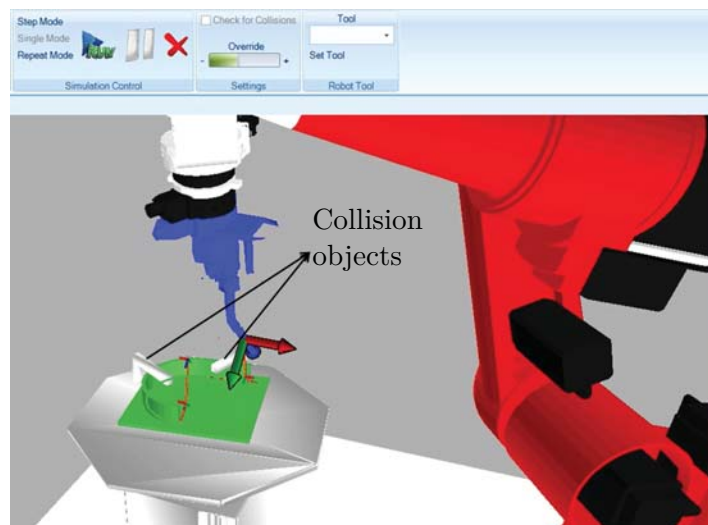
participants were asked to perform the task in the fastest and most optimal way and the optimality was also verbally explained.

The end-user had to select the desired features to be welded and to follow the macros. Both simulation environments were previously configured. The configuration consisted of the preparation of the scene and the setting of the collision detection function. For the Delfoi software, the welding process in the DelfoiArc toolbox was previously configured. Similarly, the welding process parameters in the XML format were previously configured in the internal developed simulation. Therefore in contrast to the first experiment, no configuration of the process was required in this experiment by the participants.

The time for the programming using each simulation environment was measured up to the point in time the simulation was run without collisions. Six participants performed the simulation. The average required time using the Delfoi software was 629 s. For the AOLP approach with the internally developed simulation, the average required time was 163.5 s. The given welding task for avoiding the collision was achieved 3.84 times faster using the AOLP in comparison to the commercial OLP software.



(a) Delfoi simulation environment.



(b) Internal developed simulation environment with the AOLP approach.

Figure 4.26: Simulation of the collision avoidance in two different OLP environments.

4.5.4 Close-to-Optimal Workpiece Positioning for Robotic Welding

As a further extension to the implementation, the workpiece positioning (refer to Section 3.5) for the welding process with a rotatory table is also simulated and evaluated (Diaz Posada, Mukherjee et al. 2018). Characteristics such as avoidance of welding defects due to the gravity or the facilitation of the optimal welding angle are criteria commonly defined by the process experts. The rotatory table offers two DoFs denoted as Ψ_1 and Ψ_2 as depicted in Figure 4.27a. The bounds for angle Ψ_1 are defined as 95 Deg and -95 Deg. The bounds for angle Ψ_2 are defined as 170 Deg and -170 Deg. Figure 4.27b depicts the real robot cell (Reis RV30-26 as defined in Subsection 4.5.1 in page 87).

The optimal orientation with respect to the product feature is optimized in the path planning phase as presented in previous sections. Moreover, a minimum joint movement and gravity cost are also used as criteria while positioning the workpiece by using the DoFs of the rotatory table. The path plannings are performed in this subsection for the semi-circumferential feature of the workpiece presented in Figure 4.1b in page 81.

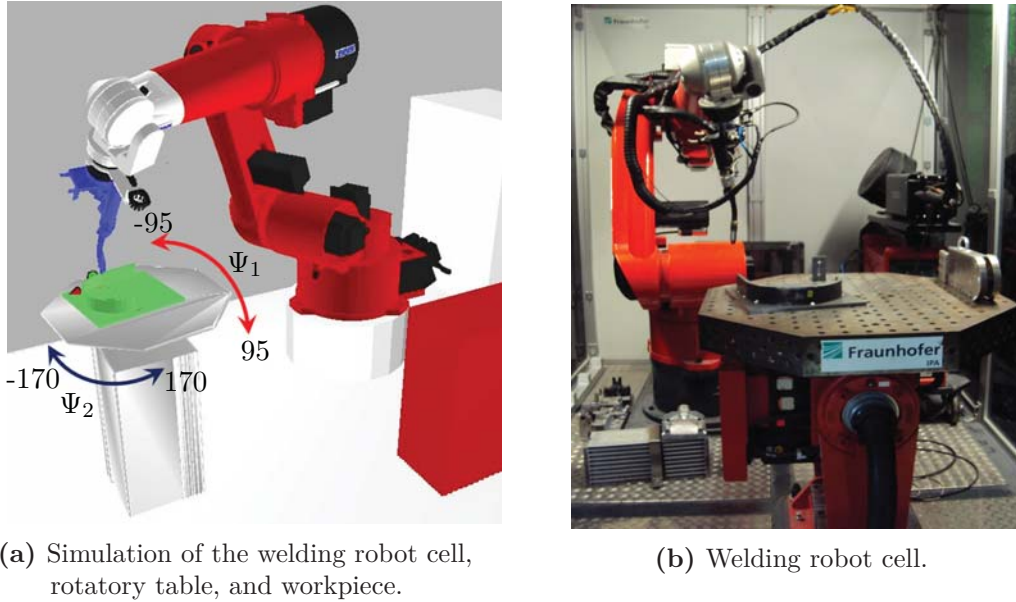


Figure 4.27: Simulation and real robotic welding cell.

The DoF of the process is defined as the rotation around the vector $\delta\alpha$, in which the welding direction is defined. This is achieved by configuring the process in the XML format. The validity functions taken into account in the workpiece optimization are the reachability and maximum joint limits functions (as defined in Subsection 3.2.1). Validity results for discretizations every 10 Deg for this use-case over the defined DoFs of the workpiece positioning problem are depicted in Figure 4.28. If the planner cannot interconnect a solution between the optimized start or goal state, or if there is not a possible goal or start, the planner reports the anomaly (red color). This anomaly is caused due to the costmap of the RMP² for the specific workpiece position.

The state cost for the welding path planning C_s is defined as in Subsection 4.3.1 and equation (4.6) in page 84. Based on this definition, the total motion cost for the welding process wC_t is defined as the integral cost over the current computed path plus the backward policy as described in Subsection 4.3.2 and equation (4.8) in page 85. The total motion cost has units in [Deg · %] and it is divided over ten in order to obtain similar ranges in comparison the other costs in this use-case.

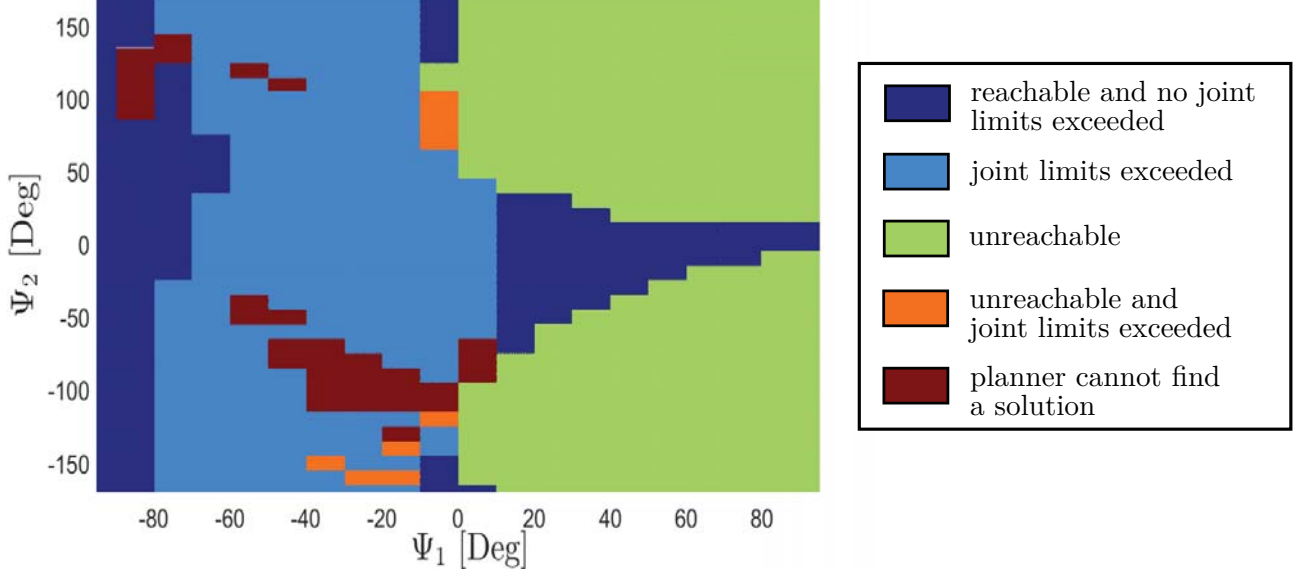


Figure 4.28: Validity evaluation for the workpiece positioning in the welding use-case. Dark blue: reachable and no joint limits exceeded; Light blue: joint limits exceeded; Green: unreachable; Orange: unreachable and joint limits exceeded; Red: planner cannot find a solution.

Note that the welding total motion cost is the only one taken into account in the path planning phase. Other later defined costs are considered during the workpiece positioning. Figure 4.29a depicts the costmap through all the feasible workpiece poses (unreachable states are not explored) after running an RRT* algorithm through the automatically set $\mathbb{RMIP}^2$ for 20 seconds in order to optimize ${}^w\mathbf{C}_t$. In this case, an initial check is done in order to evaluate if the optimum through the dimension k is feasible. Remember that the k dimension parameterizes the translation constraint of the weld seam (as defined in Subsection 4.2.1).

For the workpiece positioning, two additional cost are defined: The first one is the joint movement cost for welding (${}^w\mathbf{C}_j$) in order to prioritize less joint movements during manufacturing process execution. This functionality is useful if energy savings are desired. Each joint is assigned with a different weight w_j depending on its relevance. These weighting factors w_j are defined in the simulation taking into account the power consumption of each servo-motor. The deviations $\delta\theta$ of each joint j through all the poses h over the computed path are calculated as in Equation (4.10). Figure 4.29c depicts the costmap after the planning phase for each of the computed and feasible paths over a discretization of 10 Deg over the DoFs of the workpiece positioning.

$${}^w\mathbf{C}_j = \sum_{j=1}^6 \left(w_j \cdot \sum_{i=1}^h \delta\theta_{j,i} \right) \quad \text{with} \quad \delta\theta_{j,i} = \theta_{j,i+1} - \theta_{j,i} \quad (4.10)$$

Moreover, a gravity cost ${}^w\mathbf{C}_g$ is also defined for the welding use-case. In this process, it is not desired to weld in the direction in which gravity makes the welding material fall down over the welding-seam. In order to avoid this, a vector defined from the translation component of the product start (x_s, y_s, z_s) and goal (x_g, y_g, z_g) in $\mathbb{SE}^3$ is computed and its components are evaluated as defined in Equation (4.11). The angle Φ determines the influence of gravity over the welding process. If it is null, this means that there is no gravity influence because the z components of the product feature start and goal are on the same level. Its maximum value is obtained if $\Phi = 90$ Deg. Figure 4.29b depicts the costmap of

the gravity cost. Note that the costs considered during the workpiece positioning (i.e. ${}^w\mathbf{C}_j$ and ${}^w\mathbf{C}_g$) have units in [Deg] in comparison with the units of the total motion cost ([Deg · %]) in which the k dimension is implicit.

$${}^w\mathbf{C}_g = \Phi = \tan^{-1} \left(\frac{z_g - z_s}{\sqrt{(x_g - x_s)^2 + (y_g - y_s)^2}} \right) \quad (4.11)$$

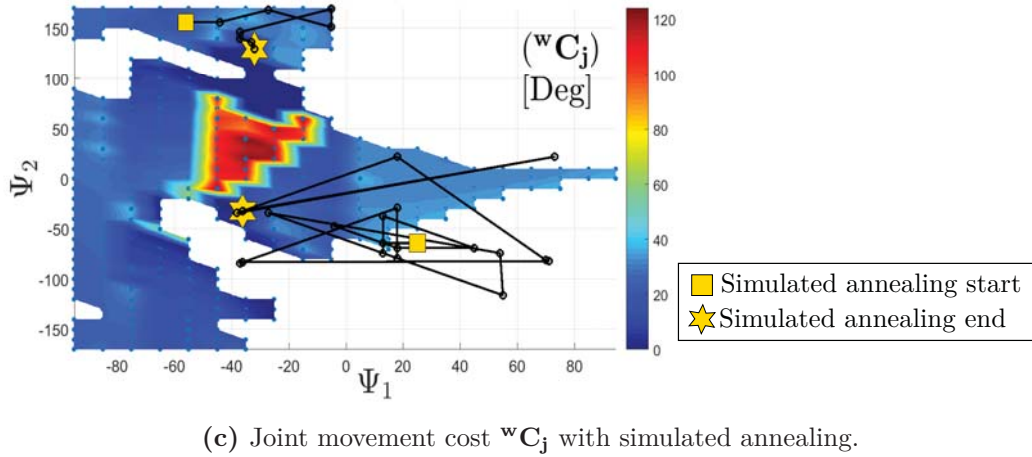
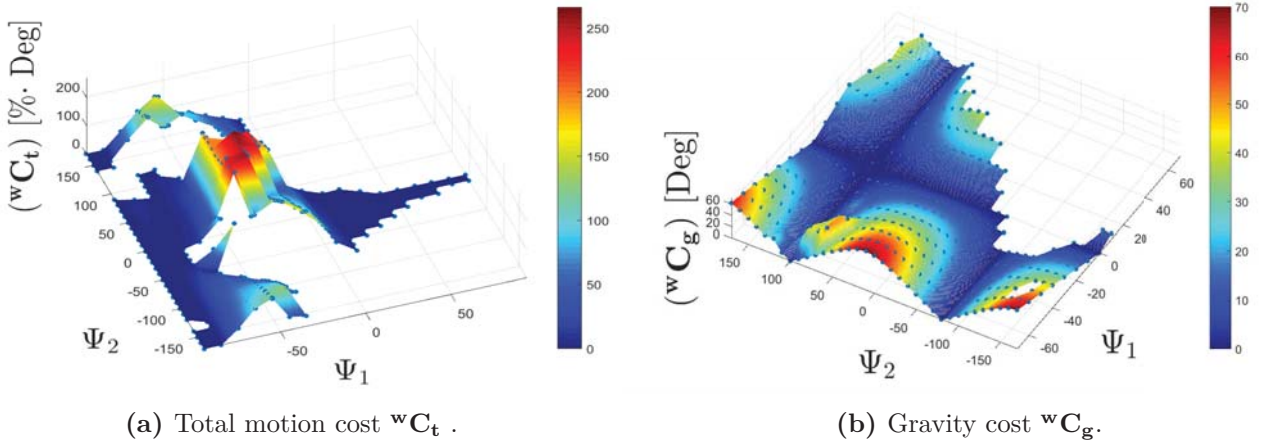


Figure 4.29: Costmaps for the workpiece positioning optimization in the welding use-case.

After these definitions, the workpiece localization cost for the welding process is defined with weighting factors w_{jm} and w_g as follows,

$${}^w\mathbf{C}_{wp} = w_{jm} \cdot {}^w\mathbf{C}_j + w_g \cdot {}^w\mathbf{C}_g \quad (4.12)$$

The simulated annealing (as presented in Section 3.5) has been run in two different configurations for optimizing the workpiece pose. The first one takes into account just the joint movement cost (refer to Figure 4.29c). Results of two simulated annealings are displayed over the costmap. The orange rectangle symbolizes the start of the simulated annealing. The orange hexagonal star, the end of the algorithm. The transformation world to workpiece T_w^{wp} is obtained by commanding the table to this configuration. The optimized path σ^* is the one in this workpiece pose. Please note that the costmap was plotted by iterating the planner and it is just performed for explanation purposes.

Programs were generated for the optimized position $\Psi_1 = -32$ Deg and $\Psi_2 = 130$ Deg (optimized solution) and $\Psi_1 = -35$ Deg and $\Psi_2 = 20$ Deg (worst solution over the costmap). The weights configured for this criterion are $w_1 = 0.22$; $w_2 = 0.32$; $w_3 = 0.175$; $w_4 = 0.12$; $w_5 = 0.12$; and $w_6 = 0.045$ which correspond to the factors of power consumption of each servomotor. Figure 4.30 depicts the joint values obtained from the robot controller after moving the robot with an automatically generated program in Cartesian space. Slightly different velocities were set manually for each program. Note that the velocities do not influence the joint movement cost. The joint movement cost as specified in Equation (4.10) for the optimized and worst workpiece position are calculated as 24.39 Deg and 94.51 Deg respectively considering the joint values obtained from the Reis controller, demonstrating the operation of the optimization.

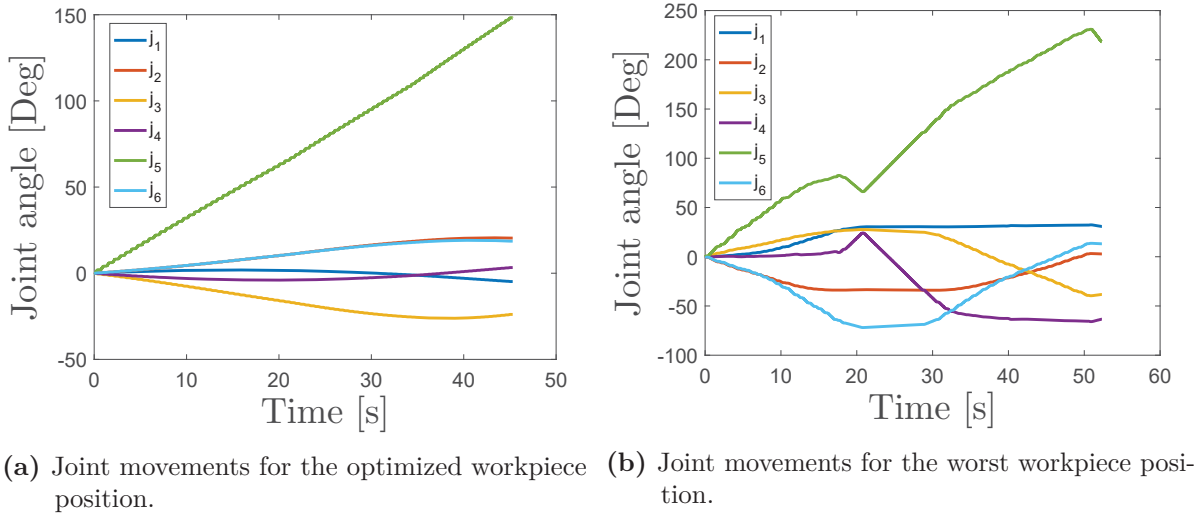


Figure 4.30: Robot joint movements for the optimized and worst welding workpiece position with joint movement optimization.

The second different configuration of the simulated annealing is done with a weighting factor of $w_t = 0.33$ (refer to Equation (3.19) in page 74), $w_{jm} = 0.33$, and $w_g = 0.33$ for depicting how the criteria could be combined and defined by the end-user. Note that w_{wp} , introduced also in page 74, is defined as $w_{wp} = w_{jm} + w_g$. The weighting factor w_t is for the total motion cost defined as in Subsection 3.5. The costmap of this configuration and results of two simulated annealings are depicted in Figure 4.31.

Figure 4.32a depicts the punctual path planning when the most optimal path is possible (no infeasibilities on the optimum). The optimized path is defined from $k = 0$ to $k = 100$ for $\delta\alpha = 0$. The samples are also displayed here for illustration purposes. Figure 4.32b depicts the punctual path planning and the algorithm avoiding the infeasible robot poses. Optimized start and goal states are found and an optimized path is found over the costmap after the exploration of the RRT*. This path has a higher cost than the optimal one. Results of the optimization in the start and goal region (S_s , S_g) can be observed here (refer to step5 in Section 3.4). Figure 4.32c depicts the planning in an area in which the path is not possible. This situation was depicted with red color in Figure 4.28. Note that in the costmap the k dimension defines the constrained semi-circumferential feature. Transparent spaces in the costmaps represent infeasibilities (in this case: unreachabilities and/or maximum joint limits). Relations of Ψ_1 and Ψ_2 can be observed between these plots and Figures 4.28 and 4.29a.

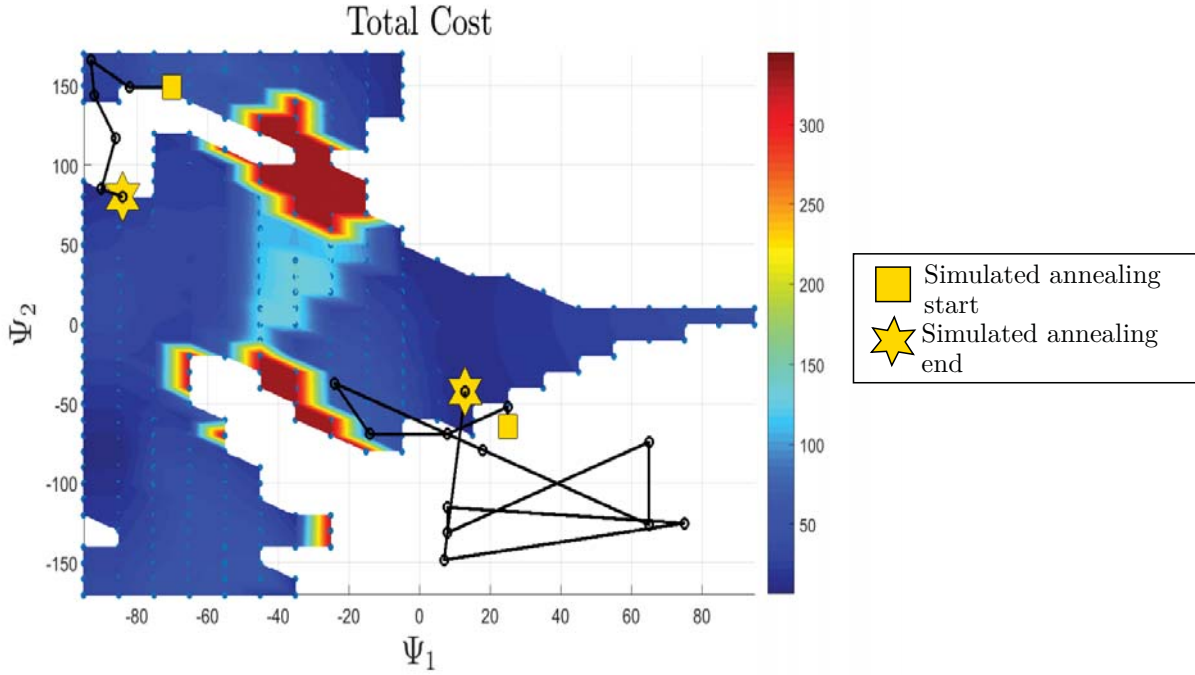
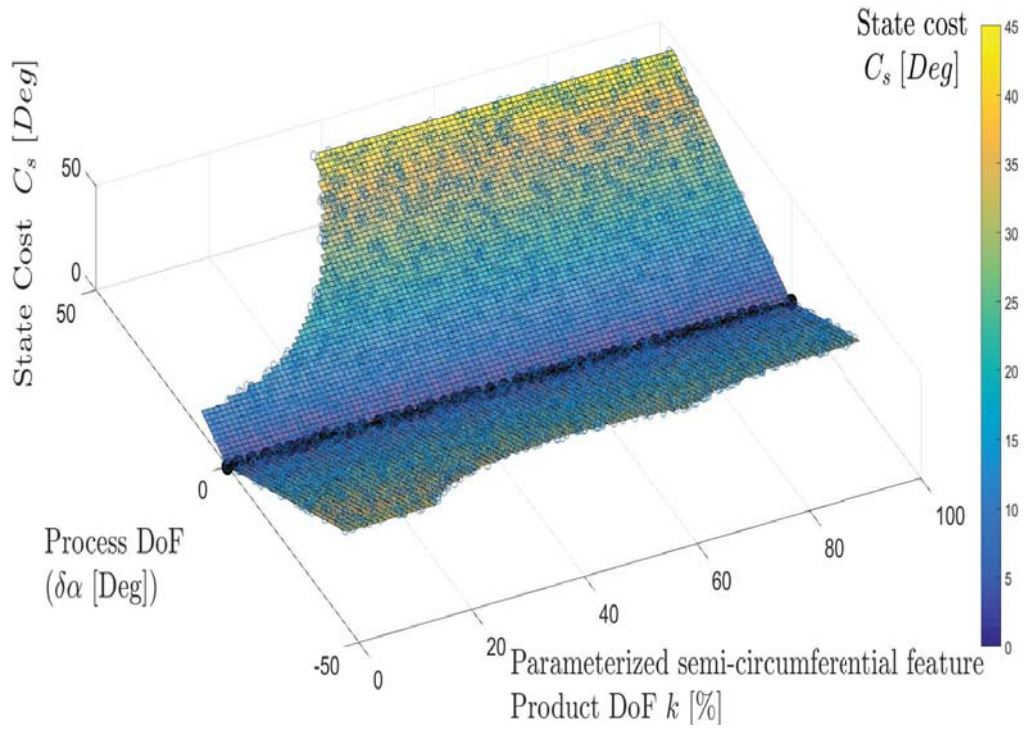
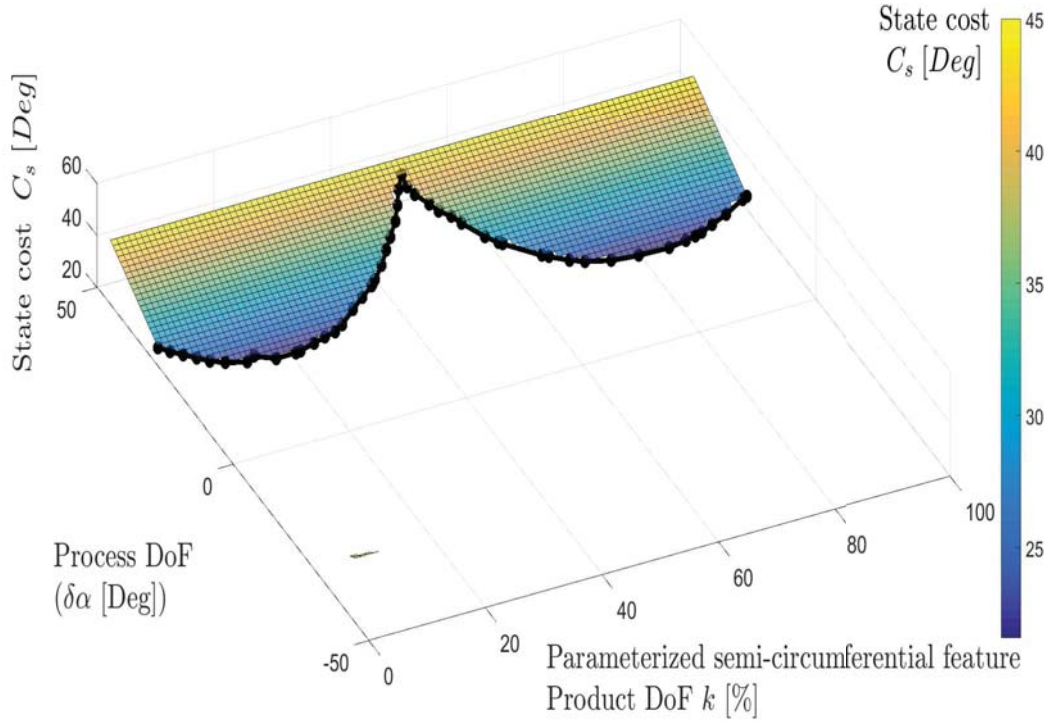


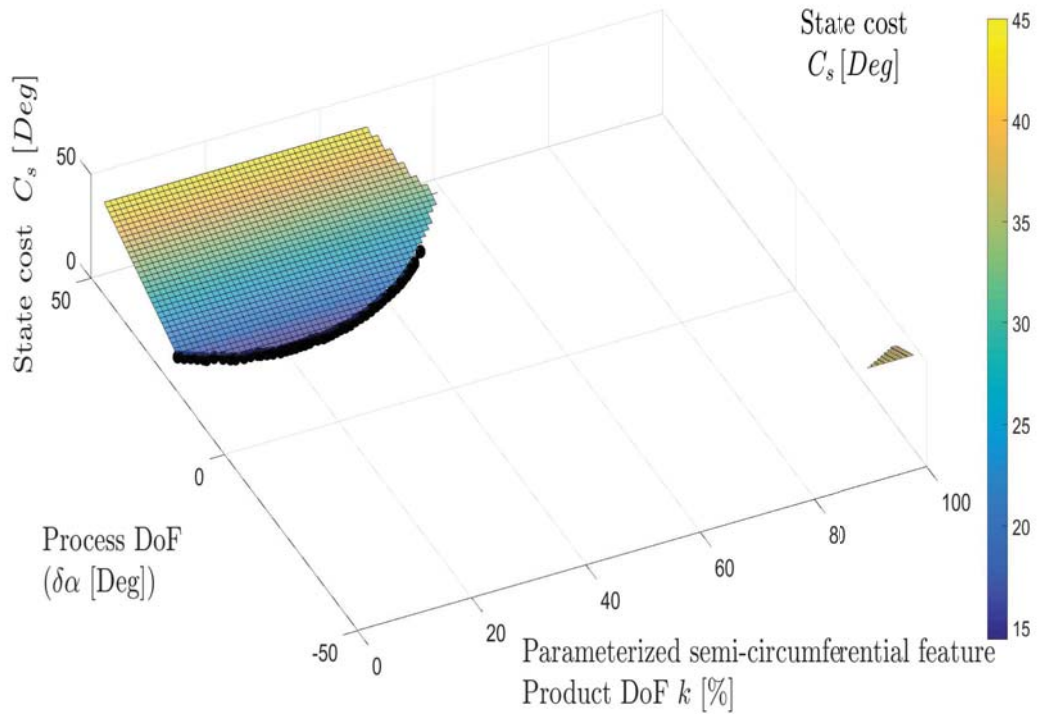
Figure 4.31: Weighted costmap for the workpiece positioning in the welding use-case with the simulated annealing algorithm and the computed close-to-optimal workpiece pose.



(a) Path planning in the costmap for optimizing welding orientation in the rotatory table configuration $\Psi_1 = 20$ Deg and $\Psi_2 = 0$ Deg with generated samples (blue bubbles). Welding optimal orientation possible through all the seam.



- (b) Path planning in the costmap for optimizing welding orientation in the rotatory table configuration $\Psi_1 = -35$ Deg and $\Psi_2 = -20$ Deg. Welding optimal orientation not possible through all the seam.



- (c) Path planning in the costmap for optimizing welding orientation in the rotatory table configuration $\Psi_1 = -30$ Deg and $\Psi_2 = -100$ Deg. Path generation not possible.

Figure 4.32: Path planning (black bubbles and lines) and costmaps for different rotatory table configurations.

4.6 Conclusion

This chapter described the use-case of the optimized collision avoidance in robotic welding. The main contribution is the performance evaluation of the proposed approach with significant improvements in terms of the required programming time and final welding quality. The avoidance of collisions is a time-consuming task using state-of-the-art methods and therefore the implementation of this use case.

In order to illustrate the modularity of the approach, the simulation has been carried out on two different workpieces with continuous features, the first with linear features and the second composed of a semi-circumferential and a linear feature. In this use-case, the state cost has been defined as the minimum required angle of a non-collision generated sample for reaching the optimal orientation for welding.

The termination algorithm, the angular velocity motion cost, and the path planning with more than one process DoF have been computed and evaluated. Moreover, a benchmark of four different sample-based algorithms was carried out. Experimentation has been performed by comparing the proposed AOLP and the teach-in approach. The programming time is on average 4.23 times faster than the teach-in approach for the given welding task. Moreover, the AOLP improved the homogeneity of the welded throat thickness by 35.62 % after experimentation with a robot cell for the evaluated use-case. The approach is on average 3.84 times faster than those implemented in a commercial OLP software for the evaluated collision avoidance. As an extension, the close-to-optimal workpiece positioning for robotic welding has been presented and validated for the minimization of joint movements.

5 Optimized Stiffness for Robotic Milling

In this chapter, the use-case of automatic robotic path generation for the milling process is described. The problematic and motivation of this use-case was defined in Subsection 2.7.2. The main difference of this use-case in comparison with the others is that this one uses models in joint space of the robot for building the costmap which defines the robotic manufacturing process. The process forces and the robot joint compliance model are used to optimize the robot stiffness. Both strategies for interpreting multiple product features (refer to Subsection 3.1.1) are simulated and evaluated in experimentation.

A similar structure as in the approach and architecture as well as in the welding use-case chapters is addressed for depicting the modularity of the approach. Section 5.1 formally defines the problem. Section 5.2 describes the product, process and resource model of this use-case. Section 5.3 introduces the state and motion cost. Section 5.4 describes the interpretation and planning of the stiffness optimized path. Section 5.5 presents the simulation and experimentation results of the computed optimized paths. Section 5.6 concludes this use-case.

5.1 Problem Definition

To address improvements in robotic machining accuracy, the following path generation problem is defined as in (Diaz Posada, Schneider et al. 2017): compute automatically and offline a feasible and optimized path σ^* optimizing the stiffness of the robot $\mathbf{f}_{\text{stiff}} = (\mathbf{f}_{\text{compl}})^{-1}$ by using the degree of freedom of the rotation around the tool axis $\delta\gamma$ throughout all the feature to be milled. The optimized path must take into account the specified end-user task definition and the product, process, and resource components specified by models as determined in the next section.

5.2 Product, Process, and Resource Modeling

The following section describes the proposed product, process, and resource models for this use-case which are the inputs required for the automatic interpretation and configuration of the path planner in order to solve the stated problem.

5.2.1 Product Model

The product model for this use-case consists of multiple continuous milling paths of a workpiece. The milling paths are selected by the end-user with a click using the simulation environment, determining in this way the task specification of the process. The end-user determines with the clicks the sequence of the machining. The software has been developed for allowing the modification of the direction of a selected machining path. The workpiece to be machined is composed of linear features.

The linear product features for this use-case have been modeled with the two different strategies presented in Subsection 3.1.1. In the first one, the single problem strategy applying the model-based piecewise approach is used for computing the transformation $T_w^p(k)$. Algorithm 1 (refer to page 54) is used for symbolically encoding this transformation defining just one planning problem ($\rho = 1$).

The multiple problem strategy is applied for the second evaluation. Here for each feature, a path generation problem is defined, being then $\rho = u$ and the $T_w^p(k)$ as defined in Equation (3.5) in page 55. Remember that u denotes the number of selected continues features. This strategy is implemented for avoiding abrupt changes of orientation in $\mathbb{SE}^3$ due to the $\mathbb{RMP}^n$'s discontinuities, presented as a result of the nature of the state cost with respect to the product features definitions. A heuristic function is implemented for assuring that the state costs and that the configurations in $\mathbb{SE}^3$ of each goal and next feature start states do not deviate significantly (refer to step 5 in Section 3.4). The optimization in the OLP (refer to Subsection 5.5.2) and the experimentation (refer to Subsection 5.5.3) will be presented taking into account these two strategies.

The rotational component for both strategies of the milling path is modeled by obtaining a quaternion denoted as q based on a defined product orientation convention. This orientation is computed taking into account the direction vector of the machining process as the $\vec{x}$ vector of the tool coordinate system (red vectors depicted in Figure 5.1) and the $\vec{z}$ vector as the direction in which the milling tool points (blue vectors depicted in Figure 5.1). The quaternion and the start and end points of each linear feature are used to compute a homogeneous transformation defining the frame attached to the workpiece feature in which its orientation match with the desired milling direction. Figure 5.1 depicts the product model and its notations for the milling process. The DoF of the product (denoted DoF_p) is represented with the variable k as in the previous use-case. Remember that as defined in Subsection 3.1.1, $\vec{V}_{sm}$ and $\vec{V}_{gm}$ are the start and goal position vectors of the features. The constant $k_d = 100$ is the end of parametrization for a single product feature.

5.2.2 Process Model

As defined in Subsection 3.1.2, the process description is performed locally, meaning with respect to the coordinate system of the machining path in the workpiece. Position and rotational constants $(x, y, z, \alpha, \beta, \gamma)_{proc}$ for this transformation are zero values because the tool needs to satisfy the constraint of the product milling path (tool in contact operation with the workpiece). The DoF of the process (DoF_{proc}) is defined as $\delta\gamma$, which is the rotation around the tool axis ($\vec{z}$ axis of the tool coordinate system as in Figure 5.1; the RGB convention- red, green, blue for $\vec{x}$, $\vec{y}$ and $\vec{z}$ respectively- is used here and through all the document). Figure 5.1 depicts the inputs required for interpreting the transformation T_p^{proc} . The following equation notates the transformation for this use-case,

$$T_p^{proc} \left(DoF_{proc} \right) = T_p^{proc} \left(\delta\gamma \right)_{proc} = \left[\mathbf{R}_{eul} \begin{pmatrix} 0 \\ 0 \\ \delta\gamma \\ 0 \end{pmatrix}_{proc} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}_{proc} \right]. \quad (5.1)$$

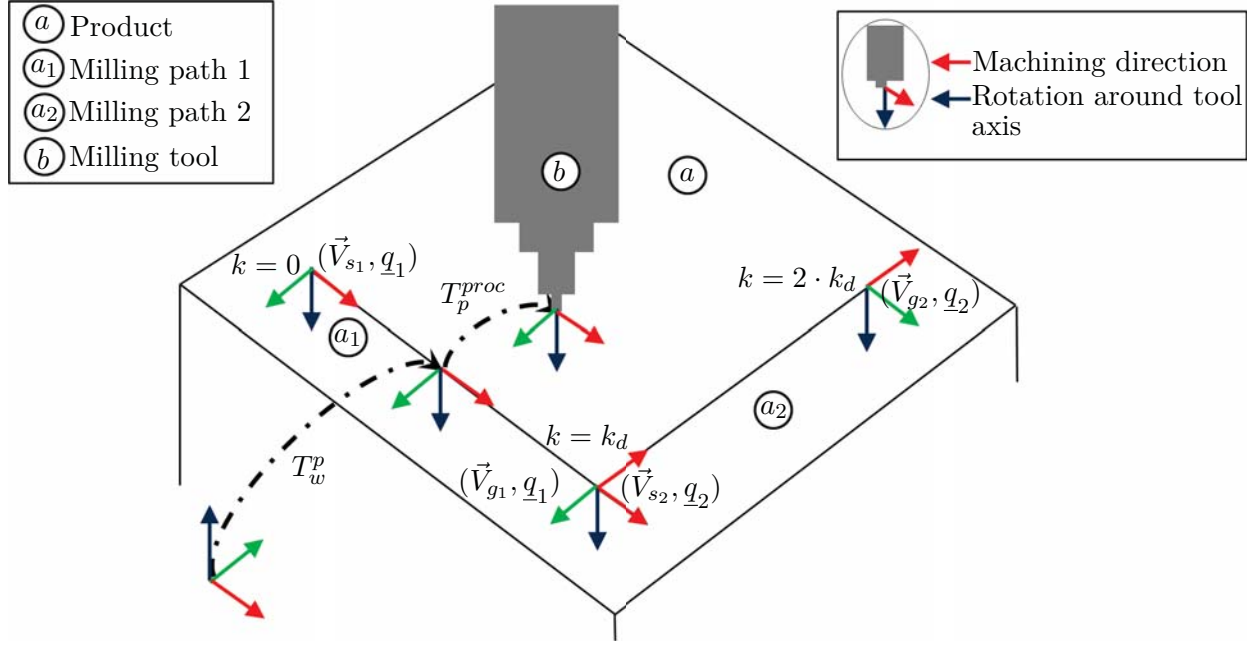


Figure 5.1: Illustration of two milling paths on the product and its notations for two linear features.

The definition of the process model parameters is done in an XML file. In this file, the process parameters are specified and the degree of freedom for the process are set. Listing 5.1 depicts the semantic model of the milling process in the XML format with process parameters, such as spindle speed, feed rate, depth of cut, number of tool teeth, and the process frame characteristics. For this use-case, also process parameters are determined because a model for computing the forces due to the milling is adopted. The computation of process forces is explained below. The calculation of robot deviations due to compliance is explained in Section 5.3. For the milling process, the minimum ($\delta\gamma_{min}$) and maximum ($\delta\gamma_{max}$) of rotation around the tool axis are defined as -180 Deg and +180 Deg due to the complete possible rotation around the vector in which the milling tool points ($\vec{z}$).

```

1<Process name="Milling" type="Edge">
2 <Parameter name="SpindleSpeed" unit="1/min" value="10000"/>
3 <Parameter name="FeedRate" unit="mm/min" value="300"/>
4 <Parameter name="DeepCut" unit="mm" value="1"/>
5 <Parameter name="ToolDiameter" unit="mm" value="8"/>
6 <Parameter name="NrToolTeeth" unit="Nr" value="4"/>
7 <FrameConstraintsDoF>
8 <GeoParameter name="x" unit="mm" type="Fixed" value="0"/>
9 <GeoParameter name="y" unit="mm" type="Fixed" value="0"/>
10 <GeoParameter name="z" unit="mm" type="Fixed" value="0"/>
11 <GeoParameter name="α" unit="degree" type="Fixed" value="0"/>
12 <GeoParameter name="β" unit="degree" type="Fixed" value="0"/>
13 <GeoParameter name="γ" unit="degree" type="Range" min="-180" max="180" value="0"/>
14 </FrameConstraintsDoF>
15 <OptimizationFunction name="Stiffness_Optimization">
16 </OptimizationFunction>
17</Process>

```

Listing 5.1: Semantic modeling of the milling parameters in XML format.

Process Force Calculation

The process model also specifies the process-dependent parameters that are used to estimate the forces experienced by the robot end-effector during the milling. These parameters are determined according to the tool producer's specifications. In this way, the feed per tooth (f_z) in mm/tooth is quantified. The feed rate V_f in mm/min of the tool is calculated using the standardized equation $V_f = n_z \cdot z \cdot f_z$. Where n_z is the spindle speed in rpm, z the number of cutting teeth, and d the diameter of the tool in mm. From this equation, f_z can be determined.

The above calculated parameters are used to estimate the process force by using the equation specified by Gomeringer (2014). The force in machining direction is calculated using the following equation $\vec{F}_{Proc} = a_p \cdot h_m \cdot K_c$ (Gomeringer 2014, p.350), where h_m is the chipping thickness, calculated with $h_m = \frac{360^\circ}{\pi \Delta \phi} \left(\frac{a_e}{d} \right) f_z$ (Gomeringer 2014, p.350). Here $\Delta \phi$ is the entry angle calculated by $\cos \Delta \phi = 1 - \frac{2a_e}{d}$. The constant a_e is the depth of cut in mm; d is the diameter of the tool in mm; K_c is the material-specific cutting coefficient, and a_p the cutting width in mm (as in (Gomeringer 2014, p.346)).

5.2.3 Robot Model: Joint Wrench Computation due to Process Force and Joint Weights

This subsection describes the models of the resource component, in this case, an industrial robot denoted as (ROBOT) with a mounted milling tool on its end-effector. These models consist of the robot forward and inverse kinematics generally described in Subsection 3.1.3, the wrench computations in joint space, and the robot joint compliance models which are explained below.

The wrench in joint space is defined in a coordinate system A as the tuple composed of the force and the moment as follows $(\omega_A) = [\vec{F}, \vec{\tau}]^T$, where $\vec{F} = [F_x, F_y, F_z]^T$ represents the forces and $\vec{\tau} = [\tau_x, \tau_y, \tau_z]^T$ the moments in the $\vec{x}$, $\vec{y}$ and $\vec{z}$ directions of frame A . For computing the wrenches, which are required for the state cost evaluation (refer to Subsection 5.3.1), the process force and joint masses with their center of masses are required as inputs. The evaluation of the process force $\vec{F}_{Proc}$ was described in Subsection 5.2.2 and the estimation of masses and the center of masses are detailed in Appendix A.

The wrench of each joint (in frame A) is computed using the transformation from its following joint or coordinate system where forces or moments are applied (in frame B) with the force-moment transformation $\mathbf{T}\mathbf{w}$ (Craig 1989, p.181), where R represents the rotation matrix and $\vec{V}$ the translation vector as follows:

$$\omega_A = \begin{bmatrix} \vec{F}_A^A \\ \vec{\tau}_A^A \end{bmatrix} = \begin{bmatrix} R_B^A & 0 \\ \vec{V}_B^A \times R_B^A & R_B^A \end{bmatrix} \begin{bmatrix} \vec{F}_B^B \\ \vec{\tau}_B^B \end{bmatrix} = \mathbf{T}\mathbf{w}_B^A(\omega_B) = \mathbf{T}\mathbf{w}_B^A \left(\begin{bmatrix} \vec{F}_B^B \\ \vec{\tau}_B^B \end{bmatrix} \right). \quad (5.2)$$

For joint 6, the computed milling force is mapped from the calibrated TCP frame (tcp). Moreover, the weight of the end-effector m_{EE} is mapped to the joint frame taking into account its center of mass cm_{EE} . Other center of masses are denoted as cm followed by the respective joint indicator.

Other wrenches for each joint (j) due to joint masses (denoted as $\underline{m}_j$ with the respective joint indicator) are computed in the same way. All joint weights have been defined in the negative $\vec{z}$ direction

in the global coordinate system (pointing in the gravity force direction). Equation (5.3) describes the computation of wrenches for a six-joint articulated robot for static poses.

$$\begin{aligned}
 \underline{\omega}_{j6} &= \mathbf{Tw}_{tcp}^{j6}([\vec{F}_{Proc}, 0]^T) + \mathbf{Tw}_{cm6}^{j6}([m_{j6}, 0]^T) + \mathbf{Tw}_{cmEE}^{j6}([m_{EE}, 0]^T) \\
 \underline{\omega}_{j5} &= \mathbf{Tw}_{j6}^{j5}(\underline{\omega}_{j6}) + \mathbf{Tw}_{cm5}^{j5}([m_{j5}, 0]) \\
 &\vdots \\
 \underline{\omega}_{j1} &= \mathbf{Tw}_{j2}^{j1}(\underline{\omega}_{j1}) + \mathbf{Tw}_{cm1}^{j1}([m_{j1}, 0]).
 \end{aligned} \tag{5.3}$$

Based on the joint wrench computation, the joint's moment tuple $\underline{\tau}_j$ is defined as the moment applied in the $\vec{z}$ direction for each robot joint as denoted in the next equation. The $\vec{z}$ vector in joint space defines the positive rotational vector of each robot joint.

$$\underline{\tau}_j(\underline{m}_j, cm_j, m_{EE}, cm_{EE}, \vec{F}_{Proc}) = [j6\tau_z, j5\tau_z, j4\tau_z, j3\tau_z, j2\tau_z, j1\tau_z]^T. \tag{5.4}$$

5.2.4 Robot Model: Joint Compliance Model

The robot joint compliance model has been identified for the Kuka KR270 R2700 ultra using the methodology described by Schneider, Diaz Posada et al. (2015). The model has been identified with one hundred measurements for each of three different measured robot poses. The effects of backlash are identified using third order polynomials. Positive and negative linear equations are used to model the compliance where no backlash is observed. Constants $s_a \dots s_h$, s_{min} , and s_{max} define these equations. The following equation notates the compliance model $\mathbf{f}_{compl}$ of a joint j_m , where m is the identifier of each joint. The input of this function is the joint moment τ_{j_m} applied in the $\vec{z}$ direction and its output is the respective joint deviation δj_m .

$$\delta j_m = \mathbf{f}_{compl}(\tau_{j_m}) = \begin{cases} s_c \cdot \tau_{j_m} + s_d & , \tau_{j_m} \in (-\infty, s_{min}) \\ s_e \cdot \tau_{j_m}^3 + s_f \cdot \tau_{j_m}^2 + s_g \cdot \tau_{j_m} + s_h & , \tau_{j_m} \in [s_{min}, s_{max}) \\ s_a \cdot \tau_{j_m} + s_b & , \tau_{j_m} \in [s_{max}, \infty) \end{cases}. \tag{5.5}$$

Figure 5.2 depicts the compliance model for each joint on its rotational $\vec{z}$ component (rotation around the joint axis) (Diaz Posada, Schneider et al. 2017). The different colors for the punctual measurements determine the three different measured poses. Two colors determine the different type of models. Green lines in the pictures determine the linear models; the red functions depict the third order polynomials. Values of the compliance models ($s_a \dots s_h$) for the six joints of the Kuka KR270 R2700 ultra are provided in Table B.1 in Appendix B. Table B.2 in the same appendix describes the lower and upper ranges s_{min} and s_{max} respectively, where the non-linearities are determined for each robot joint.

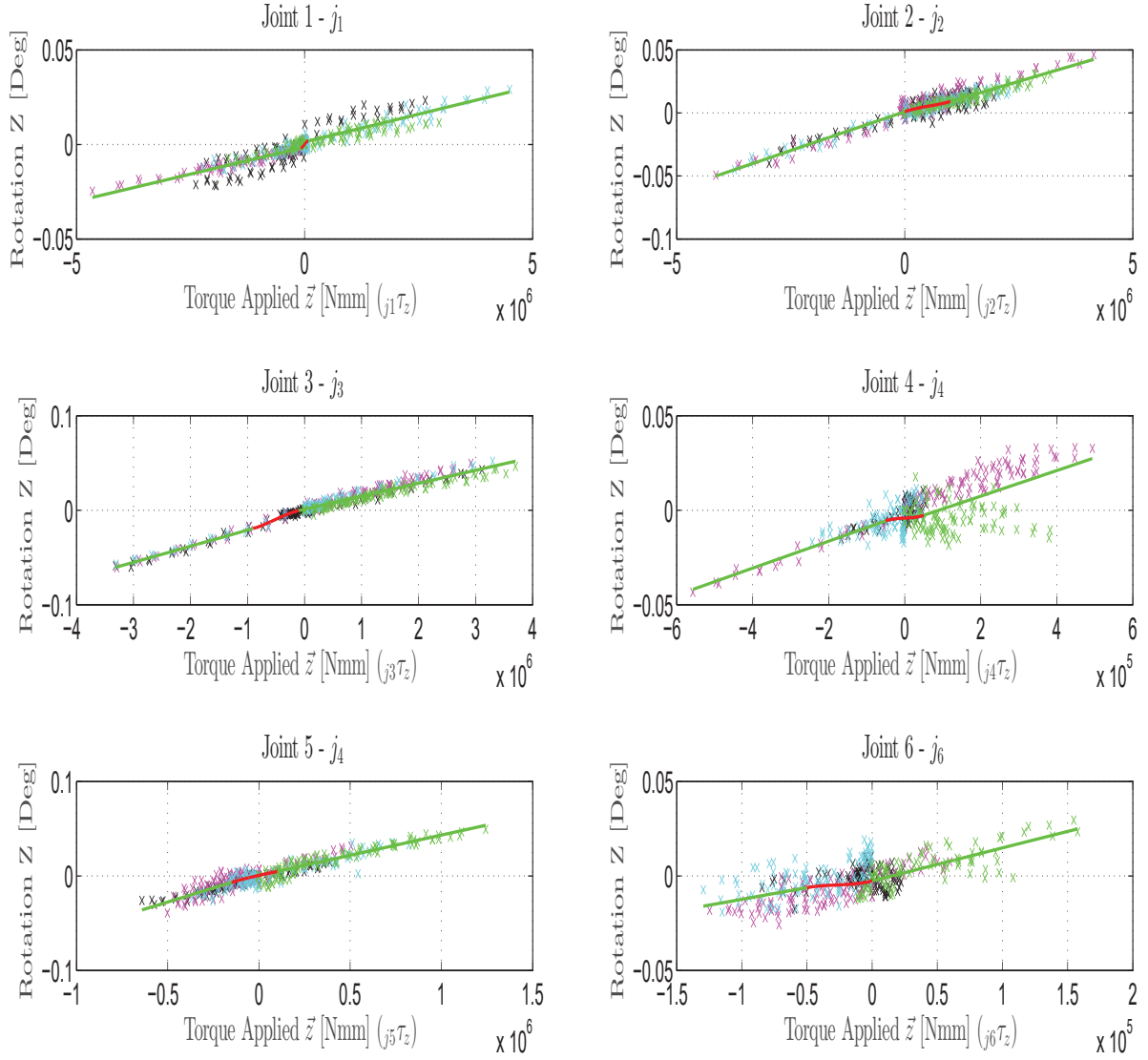


Figure 5.2: Joint compliance models for the Robot Kuka KR270 R2700 Ultra and their non-linear descriptions around the $\bar{z}$ axes of all six joints.

5.3 State and Motion Cost

5.3.1 State Cost

The state cost function $\mathbf{C}_s(\cdot)$ (generally introduced in Section 3.4) evaluates the robot stiffness in the milling process. It is defined as the translation deviations as a result of robot compliance caused by wrenches applied in each joint due to process forces and joint weights. These deviations are the Euclidean distance between the desired poses and the deviated poses caused by the robot compliance. The robot inverse kinematics is used here to transform samples from the Cartesian space to the joint space.

For calculating the robot translation deviations in Cartesian space $\underline{\delta x}$, it is necessary to first transform the samples to joint space because the robot compliance models are only available in this space. Initially, the deviations in all joints $\underline{\delta j}$ due to joint moments (τ_j , as previously defined) are computed taking into account the robot joint compliance model with the following equation,

$$\underline{\delta j} = \mathbf{f}_{\text{compl}}(\underline{\tau_j}). \quad (5.6)$$

After determining the deviations in joint space $\underline{\delta j}$, the robot translation deviations in the Cartesian space $\underline{\delta x}$ are then computed using the forward kinematics for every individual robot pose as follows,

$$\underline{\delta x} = \mathbf{FK}(\underline{j} + \underline{\delta j}) - \mathbf{FK}(\underline{j}). \quad (5.7)$$

Finally, the state cost is calculated as the Euclidean distance of the robot translation deviation in Cartesian space as follows,

$$\mathbf{C}_s(\underline{\delta x}) = \sqrt{(\delta x_x)^2 + (\delta x_y)^2 + (\delta x_z)^2} = \mathbf{C}_s(k, \delta\gamma). \quad (5.8)$$

The state cost is then a function defined for the dimension k and the degree of freedom $\delta\gamma$ of the milling process, with state costs belonging to the positive real numbers as denoted in the following equation,

$$\mathbf{C}_s(\underline{\delta x}) \rightarrow \in \mathbb{R} \geq 0. \quad (5.9)$$

Figure 5.3 depicts the described process for computing the state cost taking as inputs the product and process models for interpreting the RMP^n , then transforming RMP-samples to the Cartesian space and then to the joint space in order to calculate joint deviations caused by the the robot compliance for finally computing the Euclidean distance of the deviations which is defined as the state cost of the process.

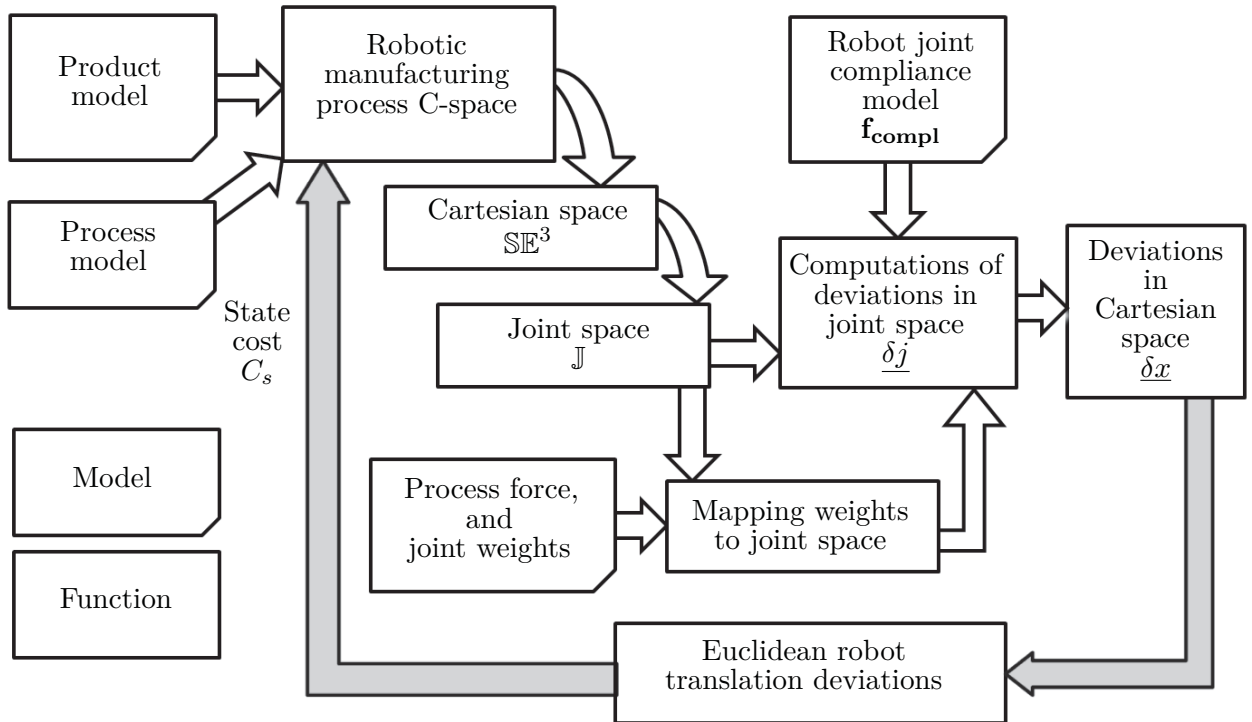


Figure 5.3: State cost computation for the stiffness optimization in robotic milling.

5.3.2 Motion Cost

Taking into account that the state cost is defined as undesired robot deviations due to its compliance characteristic and that it is desired that the motions of the robot keep the stiffest configuration for minimizing translation deviations, the objective of an appropriate path generation problem for the analyzed motion problem is defined as the computation of low-cost paths following valleys and saddle points over the RMP C-space when moving from the initial to the goal state (as in Figure 5.8 in page 127).

Considering the previous definition and the state of the art of sample-based path planners, the mechanical work motion cost (Jaillet, Cortés et al. 2016, 2008) adapted to take into account the k parameterization has been chosen as the manufacturing motion cost for solving the proposed problem. The mechanical work motion cost has been developed to compute motions that minimize the climbing of high-slope regions on elevation maps of the terrain (Jaillet, Cortés et al. 2008), which can be directly related to the proposed robot stiffness optimization problem.

The positive variations of the cost function, which in this use-case is the deviation due to the robot compliance, can be extrapolated to the proposed problem, as the movements that represent more deviations along the path. In this way, the mechanical work is adopted for the stiffness optimization problem as denoted in Equation (5.10) for a path σ , with σ_+ as portions of the path with positive slopes and with ϵ as a small penalty proportional to the distance adapted from the definition proposed by Jaillet, Cortés et al. (2008).

$$\mathbf{C}_M(\sigma) = \int_{\sigma_+} \frac{\delta C_s}{\delta k} dk + \epsilon \int_{\sigma} dk. \quad (5.10)$$

The mechanical cost in synergy with planners such as the T-RRT computes low-cost paths that follow valleys and saddle points of the configuration space costmap as required in the proposed problem. As mentioned by Jaillet, Cortés et al. (2016, p.635) “the filtering of the transition test relies on the gradient of cost function along the local motion to connect a given state to the RRT tree, resulting in an expansion biased to follow the valleys and the saddle points of the configuration space costmap”, which satisfies the requirements for optimizing stiffness.

Moreover, similar to the welding use-case, the milling process does not tolerate backward motions. Thus, the same backward policy cost $C_p = \infty$ is assigned, if the condition $(k_{s2} < k_{s1})$ is encountered (refer to Subsection 3.1.2.2). This illustrates the modularity of the approach. Based on the introduced costs, the total motion cost is then defined as $\mathbf{C}_t = \mathbf{C}_M + \mathbf{C}_p$ taking into account the definitions done in this section.

5.4 Optimized Path Generation with Stiffness Optimization

The stiffness optimization problem for the milling use-case is solved by using the general proposed approach and architecture (refer to Chapter 3). First, an interpretation of the Product, Process, and Resource (PPR) model as presented in Section 5.2 is performed. The optimization criterion of robot stiffness by minimizing deviations due to the robot compliance is also defined (refer to Section 5.3). Second, an automatic configuration of a state-of-the-art optimal sample-based path algorithm (T-RRT in this case), including its state validity definition and the state and motion cost functions is performed.

And third, the computation of an optimized motion for parsing it into a robot program, which optimizes the robot stiffness, is executed.

For this use-case, the rotation around the tool axis (refer to Subsection 5.2.2) is the degree of freedom of the process. The constraint of the product for the milling process is the path in which the milling needs to be performed. The simplification of the Cartesian space in relation to the milling process is performed via the $\mathbb{RMP}^n$ in order to reduce the computational requirements of the space sampling process. Both strategies, the single and multiple path optimization problem are evaluated in simulation and experimentation. This is summarized in Algorithm 8 with the variable m which denotes the path problem identifier.

The product and process models are interpreted in order to define the corresponding dimensions and bounds of the $\mathbb{RMP}^n$ as defined in Section 3.4. The function $\mathbf{f}(\underline{s}) : \mathbb{RMP}^n \rightarrow \mathbb{SE}^3$ is then defined as the transformation from the world to the process coordinate system ($T_w^{proc}(k, \delta\gamma)$ for this use-case). This transformation is required when evaluating the state cost presented before and when planning in the $\mathbb{RMP}^n$ for mapping samples to the Cartesian space. This transformation is specified for a milling process having as dimensions the parameterized line equation with parameter k and the degree of freedom of the tool $\delta\gamma$ as denoted in the equation below,

$$\mathbf{f}(k, \delta\gamma) = T_w^{proc}(k, \delta\gamma) = T_w^p(k) \cdot T_p^{proc}(\delta\gamma). \quad (5.11)$$

For the proposed stiffness optimization problem, two is the number of dimensions of the $\mathbb{RMP}^2$ ($n = 2$). The general function (DetermineDim (P, PROC)) is applied for determining the dimensions in software. The bounds of the $\mathbb{RMP}^2$ for this path optimization problem are defined through the function (Bounds (P, PROC)) which differs depending on the strategy selected. The minimum and the maximum value of the k dimension in the product model are assigned for the single problem strategy as defined in Equation (5.12) and for the multiple as defined in Equation (5.13) for each product feature. The minimum and maximum rotation around $\delta\gamma$ is defined as in the XML (refer to Subsection 5.2.2).

$${}^1\underline{B} = \left\{ \begin{array}{l} k, \left\{ \begin{array}{l} k_{min} = 0 \\ k_{max} = u \cdot k_d \end{array} \right. \\ \delta\gamma, \left\{ \begin{array}{l} \delta\gamma_{min} \\ \delta\gamma_{max} \end{array} \right. \end{array} \right. \quad (5.12) \quad \left| \quad {}^m\underline{B} = \left\{ \begin{array}{l} {}^mk, \left\{ \begin{array}{l} {}^mk_{min} = (m-1) \cdot k_d \\ {}^mk_{max} = m \cdot k_d \end{array} \right. \\ {}^m\delta\gamma, \left\{ \begin{array}{l} \delta\gamma_{min} \\ \delta\gamma_{max} \end{array} \right. \end{array} \right. \quad (5.13)$$

The start and goal states are defined with the minimum and maximum value of the k dimension. Note that if the single problem strategy is selected then $\rho = 1$ and therefore just one evaluation is done in Algorithm 8. For the single problem strategy, the following definitions are valid $k_s = 0$ and $k_g = u \cdot k_d$. For the multiple problem strategy ${}^mk_s = (m-1) \cdot k_d$ and ${}^mk_g = m \cdot k_d$ are assigned to each feature. For the $\delta\gamma$ dimension, the start and goal values are defined by choosing the best state cost in mk_s and mk_g (refer to step 5 in Section 3.4).

The function of finding the minimum deviation by discretization calling the state definition is denoted as ($\min(\cdot)$). In the multiple problem strategy, a heuristic method is programmed for assuring that the state cost and that the robot configuration in $\mathbb{SE}^3$ of each goal and next feature start do not deviate

significantly. This definition allows that the goal of one feature and the start of the next one in the Cartesian space present smooth robot movements. The definition of the initial and goal states for this use-case with the multiple problem strategy is denoted as follows,

$${}^m[s_s, s_g] = \begin{cases} {}^m k, \begin{cases} {}^m k_s = (m-1) \cdot k_d \\ {}^m k_g = m \cdot k_d \end{cases} \\ {}^m \delta \gamma, \begin{cases} {}^m \delta \gamma_s = \min({}^m k_s, \delta \gamma, \mathbf{C}_s, \text{IsValid}) \\ {}^m \delta \gamma_g = \min({}^m k_g, \delta \gamma, \mathbf{C}_s, \text{IsValid}) \end{cases} \end{cases} \quad (5.14)$$

The number of dimensions n is automatically configured into a `RealVectorStateSpace` object in the OMPL (Kavraki 2017). The number of dimensions for this use-case is two (being these dimensions k and $\delta \gamma$). The involved functionalities interpret this configuration for assuring proper mapping among spaces and planning. The initial and goal states ${}^m[s_s, s_g]$ are also assigned to this `RealVectorStateSpace`. Finally, a T-RRT algorithm is automatically configured.

Algorithm 8 describes the process for the automatic interpretation and computation of robotic milling optimizing stiffness using the T-RRT sample-based planning algorithm. Results for both strategies are presented in next section depicting the influence of discontinuities in the $\mathbb{RMP}^2$ and their avoidance using the described heuristic and the multiple problem strategy. The `IsValid( $\cdot$ )` function is defined as the maximum-joint limits evaluation.

Algorithm 8 Automatic interpretation and computation of the robotic milling process for optimizing stiffness using a T-RRT algorithm.

Input: P (Equation (3.4) or (3.5) depending on the interpretation strategy), $PROC$ (Equation (5.1)), $ROBOT$ (as defined in Subsection 5.2.3), $\mathbf{C}_t$ (as defined in Section 5.3), `IsValid`, ρ

Output: σ^* {stiffness optimized path for the desired milling process}

- 1: **for** $m = 1$ to ρ **do**
 - 2: ${}^m \mathbf{f}(k, \delta \gamma) = {}^m \mathbf{f}(k, \delta \gamma) \leftarrow \text{RMP}_{\text{toSE}^3}({}^m P, PROC, ROBOT)$
 - 3: $[{}^m s_s, {}^m s_g] \leftarrow \text{StartGoalStates}({}^m P, PROC, ROBOT, \mathbf{C}_s, \text{IsValid})$
 - 4: ${}^m \underline{B} \leftarrow \text{Bounds}({}^m P, PROC)$ as in Equation (5.12) or (5.13)
 - 5: ${}^m n \leftarrow \text{DetermineDim}({}^m P, PROC)$
 - 6: {sample and calculate optimized path for each feature}
 $\sigma^* \in [{}^m k_s, {}^m k_g] \leftarrow \text{T-RRT}(\mathbb{RMP}^2({}^m \mathbf{f}, n, {}^m \underline{B}), {}^m s_s, {}^m s_g, \mathbf{C}_t, \text{IsValid})$
 - 7: $m = m + 1$
 - 8: **end for**
 - 9: **return** $\sigma^* \in [{}^1 k_s, {}^\rho k_g]$ {retrieve concatenated optimized path of all features}
-

Note that the discontinuities are present in the $\mathbb{RMP}^2$ due to the changes of the product coordinate system orientation for each feature, which represents different robot joint configurations and therefore translational deviations. The final program is improved in this use-case by adding a pose in the corners for assuring the accomplishment of all product constraints.

5.5 Optimization in OLP Software and Experimentation

The following section presents the computation of robot stiffness optimization for a milling task in the OLP software and the experimentation in the robot cell introduced in Subsection 5.5.1. The optimization and experimentation is categorized into the two different strategies used for the interpretation of the product and the path planning problem (the single and multiple problem strategy, refer to Subsections 5.2.1, 5.4, and 3.1.1).

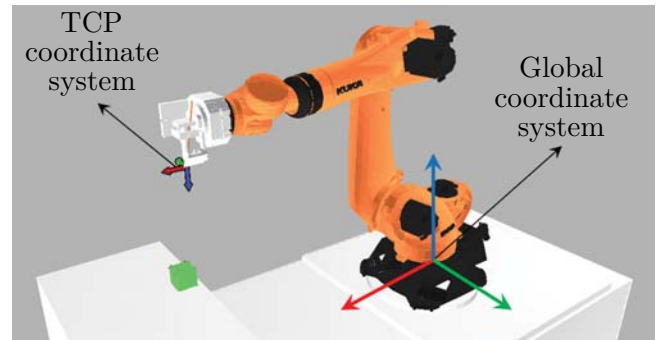
Subsection 5.5.2 presents the path optimization in OLP software and the generation of paths used for comparing final milling accuracies. Main goal of the robot stiffness optimization is to address improvements in the robotic machining accuracy. With the generated paths, milling experimentation (refer to Subsection 5.5.3) is performed in order to evaluate the final accuracy of the milled contours and their depth deviations by comparing the optimized and non-optimized paths. Subsection 5.5.4 presents the close-to-optimal workpiece positioning for robotic milling.

5.5.1 Experimental Setup

The robotic machining system consists of a Kuka Quantec KR270 R2700 ultra industrial robot, an end-effector with a Chopper spindle tool reference 3300 H S5 from the company Alfred Jäger GmbH (Ober-Mörlen, Germany) (refer to Figure 5.5). An RSP tool changer reference TC 480-1 is used to exchange different end-effectors. The robot is mounted in a machine bed of 14 tons with a table and a mechanical gripping system to hold the workpiece. Figure 5.4a depicts the described components. A 2D laser scanner and a laser tracker AT-901 is used in the experimentation (refer to Subsection 5.5.3) in order to localize the workpiece in the work-cell. For automatically generating the stiffness optimized robot program for the machining process, the robot cell has been modeled in the internally developed simulation software (refer to Section 3.6) in which the presented optimization has been implemented. The simulation of this robot cell is depicted in Figure 5.4b.



(a) Robot milling cell. Photo (Fraunhofer IPA, Rainer Bez).



(b) Simulated robot milling cell.

Figure 5.4: Real and simulated robotic milling cell.

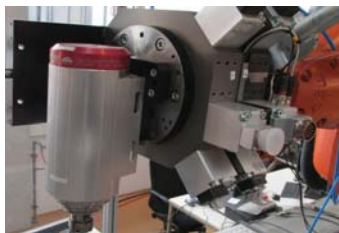


Figure 5.5: Robot end-effector with the milling tool.

5.5.2 Optimization in OLP Software

The end-user has the option in the internally developed OLP software to define the desired milling paths by clicking and to re-orientate the product coordinate frame, in case the opposite machining direction is desired. The convention of the product frames is defined as in Subsection 5.2.1. Figure 5.6 depicts the CAD model of the test workpiece with four selected linear machining paths ($u = 4$) constituting a square. The programs generated for the experimentation (refer to Subsection 5.5.3) follow the sequence between corners from A to B, B to C, C to D, and D to A, as depicted in Figure 5.6. Red and green vectors represent the $\vec{x}$ and $\vec{y}$ direction of the product frame. These vectors are depicted at the end of each linear feature. The selection of the features by the end-user consists of the denominated task definition, which is interpreted taking into account the product and process models as defined in Section 5.4.

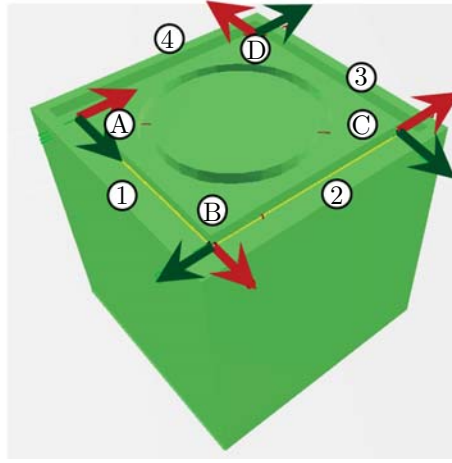


Figure 5.6: STL model of the real workpiece with 4 selected edges and its corners A, B, C, and D.

The software offers the possibility to generate paths without optimization. This path generation is based just on the geometrical specifications. For all linear features, a path has been generated without optimization. Figure 5.7 depicts the sequence of robot movements for a four-edge machining task without optimization following the product convention (comparison path for the single problem strategy). Note that due to its definition, the $\vec{x}$ vector follows the milling path. Due to the infeasible robot kinematics, the fourth path applies the rule to go in opposite direction of the $\vec{x}$ vector of the product frame. Moreover, the software also allows the generation of paths with constant orientation through all the features (comparison path for the multiple problem strategy). Different comparison paths has been approached in the experimentation for each strategy for resembling the rotational movements of the optimized paths and their consequent inaccuracies due to robot calibration.

If the optimization function is selected, the overall process of automatic configuration of the T-RRT is performed using the methodology described in Section 5.4. The result of the sampling-based process is a robot motion σ^* in the RMP^2 with optimized stiffness. The T-RRT algorithm was parameterized with an initial temperature of 0.01, a range of 7, temperature change factor of 0.1, and set frontier node ratio of 0.1 after experimentation with the algorithm. These parameters are defined in the source code of the T-RRT algorithm in OMPL. The ϵ factor of the mechanical work motion cost, as defined in Equation 5.10 in page 122, has been parameterized to 5×10^{-5} . Euclidean robot translation deviations (state cost) are retrieved in mm for the optimization and experimentation. Simulation results for the two different implemented strategies are described below.

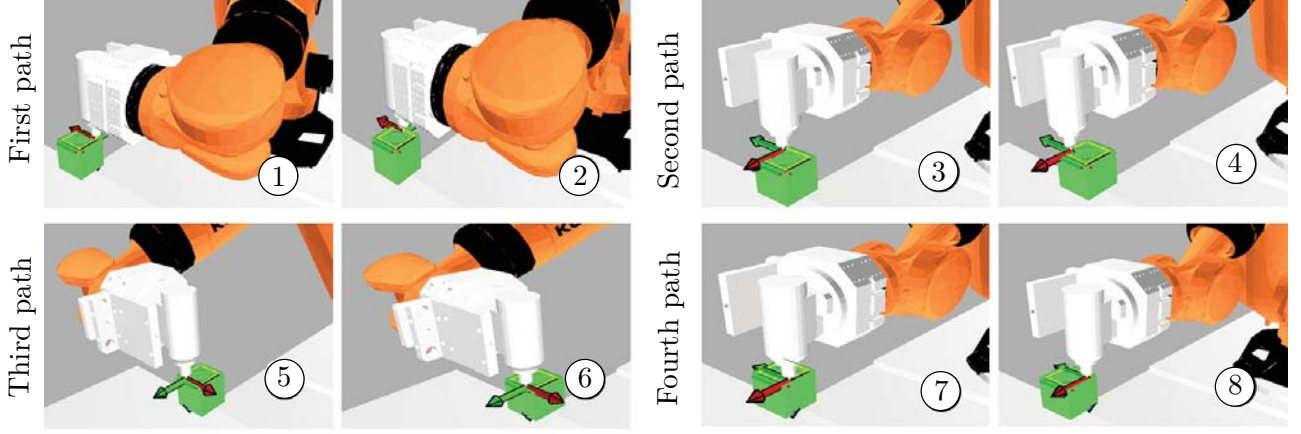


Figure 5.7: Sequence of robot movements for a four-edge machining task without optimization following the product convention (comparison path for the single problem strategy). Note that due to the definition, the $\vec{x}$ vector follows the milling path.

Optimization in OLP with the single problem strategy

After defining just one path generation problem ($\rho = 1$) and interpreting the four selected edges into one piecewise equation and planning over the $\mathbb{RMP}^2$, an optimized path is generated. Figure 5.8 depicts the state cost deviation elevation map and the planned path from the start to the goal state ($k_s = 0$ and $k_g = u \cdot k_d$, respectively) (Diaz Posada, Schneider et al. 2017). In this figure, the infeasibilities (i.e. max. joint limits) are not depicted to illustrate the overall C-space. Note that the costmap is just generated to illustrate the $\mathbb{RMP}^2$. During planning, just samples of this C-space are required. As defined in Equation (5.14), the start and goal states have been optimized by minimizing the state cost. k_d has been set as 100 as defined in Subsection 3.1.1.

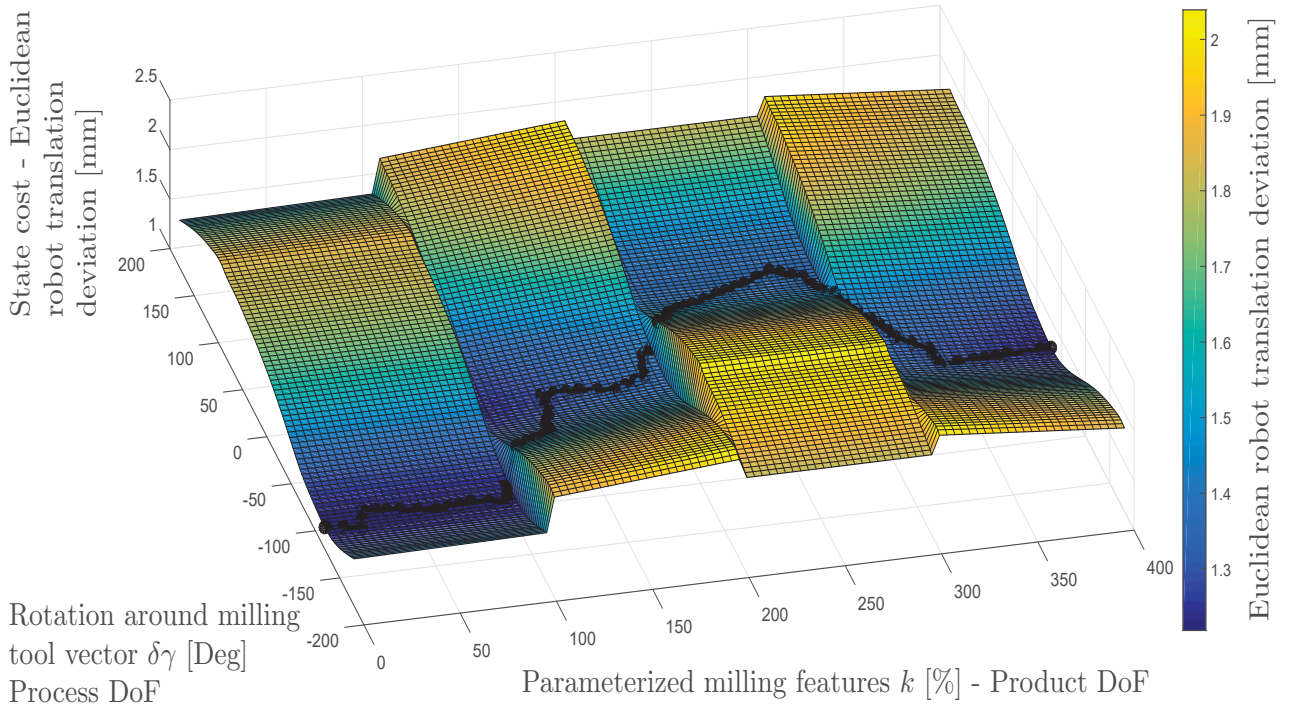


Figure 5.8: State cost (Euclidean robot translation deviation) elevation map of the path generation problem for optimizing stiffness in robotic milling and the optimized path (black bubbles and lines) with the single problem strategy.

When representing changes of features in $\mathbb{RMP}^2$ in the product dimension k , there are discontinuities in the elevation map, which corresponds to the multiples of the parameter k_d . At this stage in the $\mathbb{RMP}^2$, the definition of the product changes as defined in Algorithm 1. The change due to this discontinuity informs the path planner algorithm of transitions on the $\mathbb{RMP}^2$. Therefore, in order to prefer less cost over short paths, the parameter ϵ (refer to Equation (5.10)) has been set low in relation to the motion cost values. This allows the algorithm to find feasible paths with lower costs also through the discontinuities.

The optimized milled path in the $\mathbb{RMP}^2$ used as an automatic generated final robot program for the experimentation is illustrated in Figure 5.9. The optimized path in $\mathbb{SE}^3$ in the simulation environment is depicted in Figure 5.10. The generation of this motion using the T-RRT has been achieved in less than ten seconds. The optimized path converges to the valleys in which the robot stiffness is maximized in a slightly different way than the solution presented in Figure 5.8. This slight difference is due to the random sampling characteristic of the sample-based path planning algorithms used in this research.

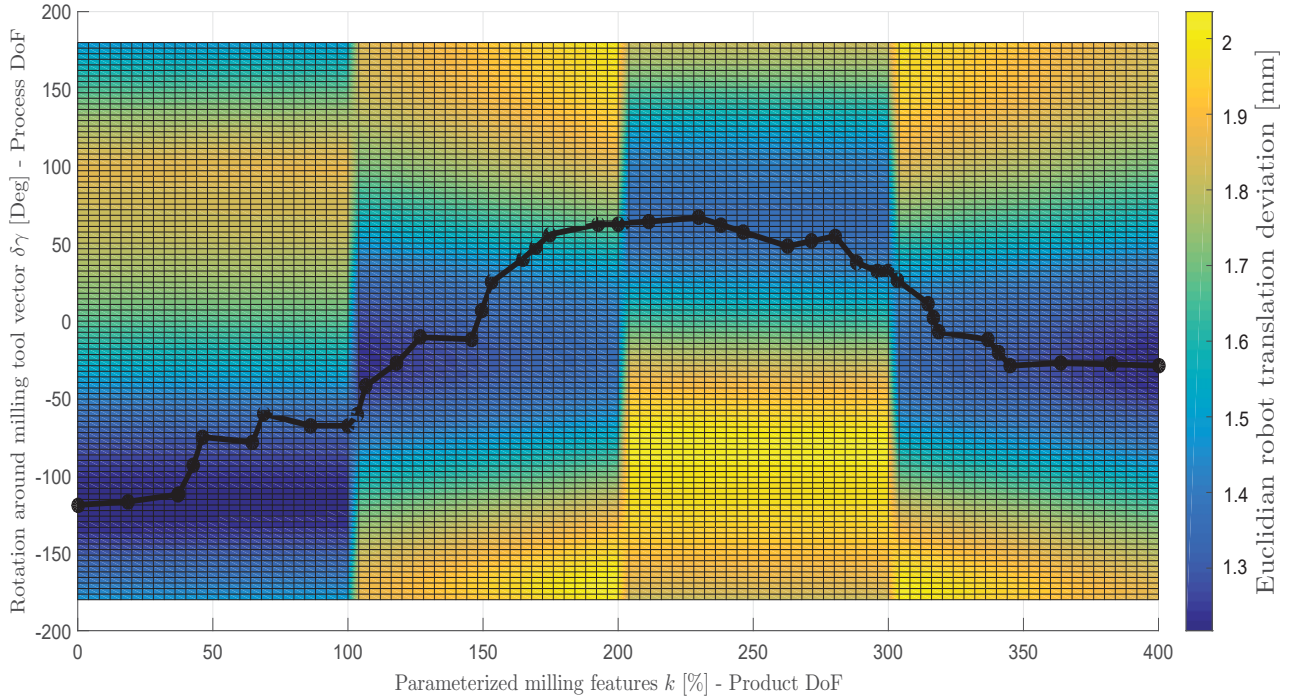


Figure 5.9: $\mathbb{RMP}^2$ for the four milled paths with their state cost (Euclidean robot translation deviations) and a computed optimized path using the T-RRT algorithm with single problem strategy.

Table 5.1 lists the required path planning times for computing the result shown in Figure 5.9 and Table 5.2 summarizes the results and characteristics for the sampling and optimization processes in the robotic milling use-case optimizing stiffness for the 4 linear features using the single problem strategy. Configurations of the T-RRT and the mechanical work motion cost were set as previously defined. Note that in comparison with the RRT* results presented in Subsection 4.5.3, there are no values for the initial total manufacturing cost, the rewires, and the heuristic iterations. This is because, per definition, the T-RRT optimizes during the search through the goal state. No further optimization is done after connecting a path from the start to the goal state. This being the reason why the automatic

termination algorithm (refer to item 10 in Section 3.4) is not used when setting the T-RRT as planning algorithm. Required times for the "plotting of space" and "sampling and optimization" are less than for the welding use case (refer to Subsection 4.5.2) because the collision detection takes longer to be computed and the T-RRT sample less than the RRT*. Milling results obtained with this optimized path are presented in the next subsection.

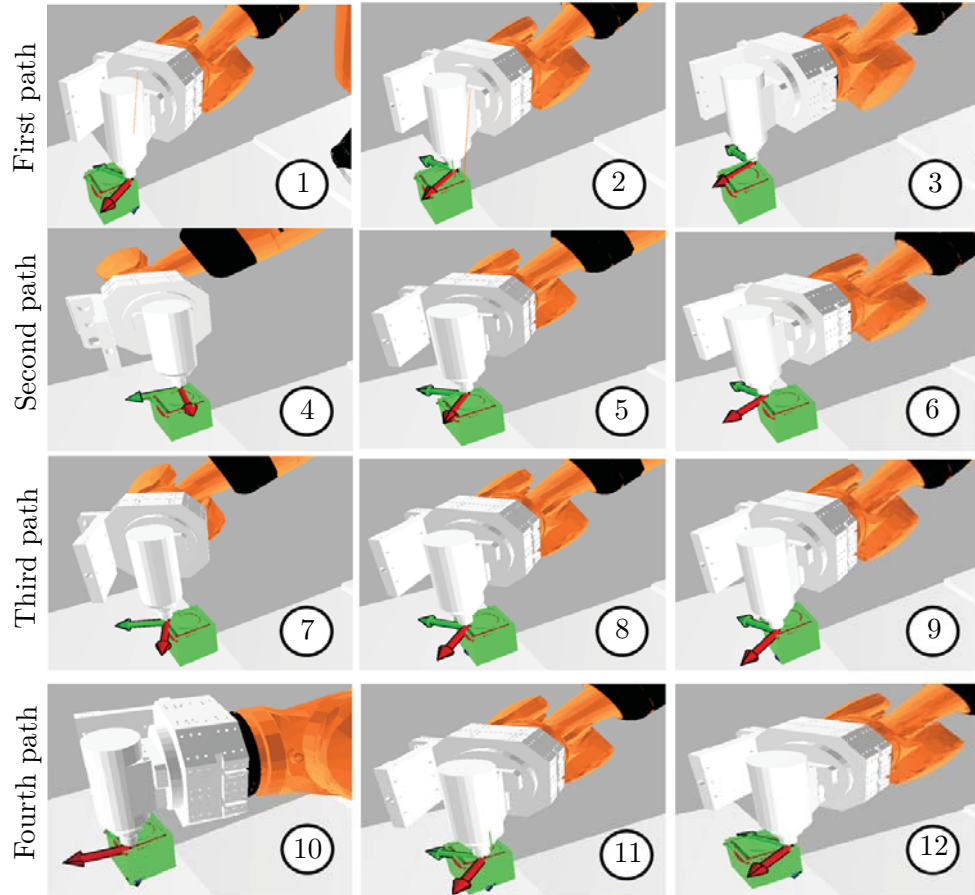


Figure 5.10: Sequence of robot movements for a four-edge machining task with optimized stiffness after generating the path with the T-RRT algorithm using the single problem strategy.

Table 5.1: Required path planning times for the robotic milling use-case optimizing stiffness for the four linear features using the single problem strategy.

Product feature	Interpretation [s]	Plotting of space [s]	Sampling and optimization [s]	Transformations [s]	Total [s]
All linear features	5.09	0.43	0.83	0.74	7.09

Table 5.2: Sampling, optimization results, and characteristics in the robotic milling use-case optimizing stiffness for the four linear features using the single problem strategy.

Product Feature	Final total manufacturing cost [mm]	Evaluated States Nr.	States in solution Nr.	Duration for sampling and optimization [s]
All linear features	0.515	316	41	0.83

Optimization in OLP with the multiple problem strategy

A robot motion was also optimized using the multiple problem strategy ($\rho = u$) and the heuristic for not allowing representative rotational deviations in the Cartesian space as detailed in Section 5.4. This strategy is conceived for avoiding the discontinuities over the $\mathbb{RMP}^2$. The path generated after planning is depicted in Figure 5.11. The OMPL simplification algorithms for interpolating and reducing vertices were used after the sample-based planning phase (refer to step 11 in Section 3.4).

Due to the planning in a continuous costmap for each feature the optimized path is more easily found than with the previously described strategy. Even though in the $\mathbb{RMP}^2$ the path presents discontinuities, when this solution is mapped into $\mathbb{SE}^3$, the discontinuities are not present due to the implemented heuristic which constrains the start state of each product feature closely to the previous goal state. Figure 5.11 depicts the path planning with the multiple problem strategy for the optimized milling task. Slight changes of the tool orientation in $\mathbb{SE}^3$ are computed due to the optimization. Figure 5.11 also depicts the path in the $\mathbb{RMP}^2$ used for comparison (constant orientation in $\mathbb{SE}^3$ with non-optimal stiffness configuration).

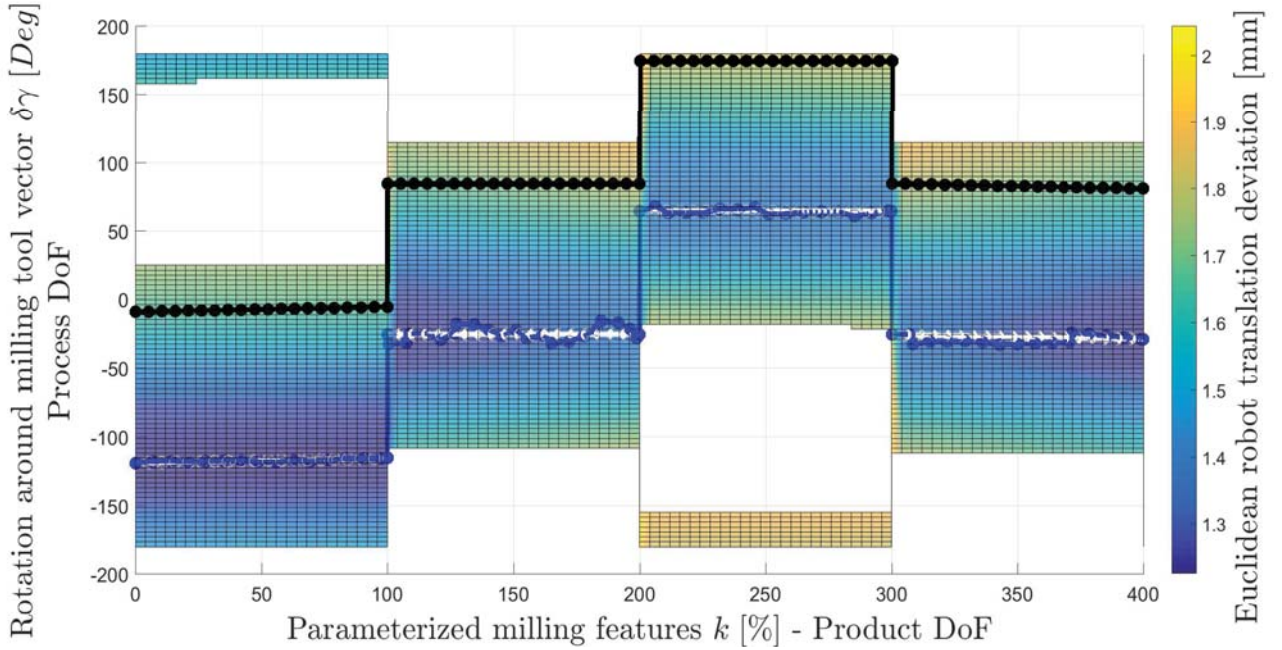


Figure 5.11: $\mathbb{RMP}^2$ for the four milled paths with their state cost (Euclidean robot translation deviations) with the computed optimized path using the T-RRT algorithm and the multiple problem strategy (blue dots and lines), the simplified path (white dots and lines), and the path used to compare the milling accuracy (black dots and lines).

Table 5.3 lists the times needed for computing the results shown in Figure 5.11 and Table 5.4 summarizes the acquired results and characteristics for the sampling and optimization in the robotic milling use-case optimizing stiffness for the 4 linear features using the single problem strategy. Configurations of the planner and the mechanical work motion cost were set as previously defined. The states in the solution are always 20, because the interpolation was activated and set in this way. Note that the final total motion cost is less using this strategy than the single problem strategy due to the avoidance of discontinuities. It can be also observed, that the final total motion cost of the 2nd linear feature is

higher than the other ones. This is due to the most relevant slope of this feature over the $\mathbb{RMP}^2$ with respect to the others as it can be noticed in the previously shown Figure 5.8. Milling results obtained with this optimized path are presented in the subsection below.

Table 5.3: Required Path planning times for the robotic milling use-case optimizing stiffness for the four linear features using the multiple problem strategy.

Product feature	Interpretation [s]	Plotting of space [s]	Sampling and optimization [s]	Transformations [s]	Total [s]
All 4 linear features	4.29	0.53	1.73	0.48	7.03

Table 5.4: Sampling and optimization results and characteristics in the robotic milling use-case optimizing stiffness for the four linear features using the multiple problem strategy.

Product Feature	Final total motion cost [mm]	Evaluated States Nr.	States in solution Nr.	Duration for sampling and optimization [s]
1 th linear feature	0.014	18	20	0.55
2 nd linear feature	0.152	28	20	0.44
3 rd linear feature	0.010	18	20	0.45
4 th linear feature	0.005	33	20	0.29
Total	0.181	97	80	1.73

5.5.3 Experimentation

Experimental evaluation of the described approach and optimizations in the OLP software were performed for both strategies using the robot cell described in Subsection 5.5.1. The robot milling tool was calibrated as described in Appendix C. Four continuous milling paths composing a square as depicted in Figure 5.6 in page 126 were milled using an automatically generated program with the optimized path and without optimization for both strategies as described above.

The machined workpiece consists of a block of steel ST-37. The four machined paths are the sides of a square of 8.6 cm over the upper face of the steel cube. The configured depth is 1 mm. The yield strength of the steel is 235 N/m^2 . A solid carbide slot drill HPC tool has been used for these milling tasks. The tool has a diameter of 8 mm, and it is from the company Horex. The tool has four teeth and a corner chamfer width at 45 Deg of 0.2 mm. The milling was performed with a rotatory speed of 10000 rev/min . The feed rate was programmed as 0.005 m/s . Entering the material (first movement in $\vec{z}$ direction), before starting the milling in $\vec{x}$ direction, was programmed with a feed rate of 0.001 m/s . Information given before is valid for all the milled paths. Results obtained in the experimentation are described categorizing both different strategies.

Experimentation with the single problem strategy

The 2D laser scanner reference scanControl 2900-50 from the company Micro-Epsilon (Ortenburg, Germany) was used for localizing the workpiece in the robot workspace relative to the robot base coordinate system for the two milled squares. The workpiece CAD model is used to generate a corresponding reference point cloud in order to locate the workpiece using the ICP algorithm by

comparing the reference with a point cloud obtained with the sensor (Kuss, Drust et al. 2016) (the localization is not part of this dissertation).

The translation deviations δx_x , δx_y and δx_z of the computed path based on the models are depicted in Figure 5.12. The most representative deviations are in the z direction due to the joint weights acting in this direction. The path optimization improved deviations in this direction as demonstrated below. In this figure, the parametrization of the four edges is performed as described in Algorithm 1 in page 54. The segment A to B is parameterized as $[0, 100)$, B to C as $[100, 200)$, C to D as $[200, 300)$, and D to A as $[300, 400]$ automatically after the task definition.

Figure 5.13 depicts the two machined squares. In the x - y plane, the inner and outer contours visualize that the optimized path is not more accurate. These deviations were expected due to the lack of absolute positioning accuracy of the robotic system while changing the TCP orientation during milling, which is the case of the optimized path. Deviations of the complete kinematic chain up to the TCP are up to 0.35 mm in x and 0.5 mm in y direction, due to rotations around z ($\delta\gamma$) as described in Appendix A. Some of the deviations in the machining are focused in Figure 5.13a to illustrate the milling inaccuracies in the x - y plane of the optimized milled path.

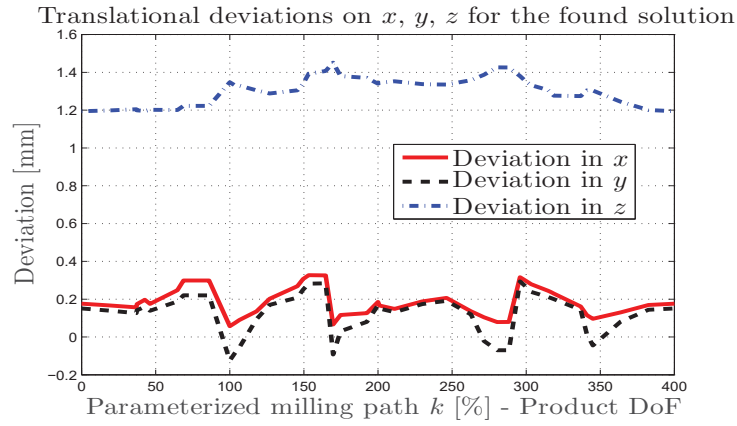
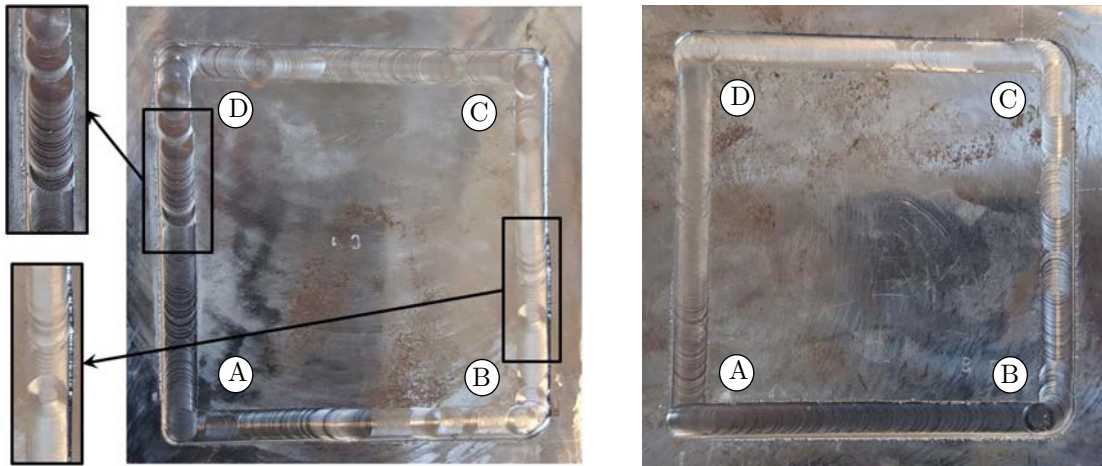


Figure 5.12: Translational deviations on x , y , z for the found solution based on the models.



(a) Machined workpiece with optimization.

(b) Machined workpiece without optimization.

Figure 5.13: Machined workpieces with corners A, B, C, and D using the single problem strategy with and without optimization.

For analyzing the milled paths, the inner contour of the square sides was measured with a Coordinate-Measuring Machine (CMM) Videocheck HA400. The 2D geometry of the milled paths is detected using the camera of the machine. By a suitable parametrization of the bright and the dark field, the 45 Deg phase over 0.2 mm of the tool can be detected. Several images are interposed by the machine in order to obtain the 2D data, which allow measuring the square edges. Based on this measurement, lines are fitted over the square sides without taking into account places where noisy measurements are observed in order to evaluate the parallelism between edges. Moreover, the same CMM machine was used with a surface feeler in order to measure the depth with respect to the upper plane of the steel cube.

The path in which the depth was measured was programmed with a 4 mm offset, half of the tool diameter, with respect to the fitted lines of the squares (between the internal and external square measured by the CMM). The parallelism of the sides of the optimized mill are not better than the one with the non-optimized path. Angle deviations between the parallel sides for the non-optimized and optimized square are -0.099 Deg and -0.417 Deg, respectively. For the perpendicular sides, the angle deviations are -0.02 Deg and -0.66 Deg, respectively.

The effects of the optimization has been verified in the z direction where depth deviations were improved. Figure 5.14 depicts the measured depth deviations for the paths with and without optimization. In order to overcome and neglect as much as possible workpiece localization errors, the first measured depth for both millings are defined as the zero deviation. The errors for the first depth with respect to the programmed milling depth (1 mm) are -0.33 mm and -0.75 mm for the optimized and non-optimized path, respectively. This error is attributed to the inaccuracies of the workpiece localization and therefore the referencing with the first measured depth.

In Figure 5.14, the same parametrization in k of the milling path as in Figure 5.12 applies. Comparing Figure 5.14 and Figure 5.9, it is observed that when the planner performed significant changes of orientation such as in the segment from 100 to 200 (B to C) the depth accuracy also reduced significantly. Improvement in the milling depth is noticed in the optimized path. The mean absolute deviation through all the path was computed for the optimized and non-optimized paths as 0.312 mm and 0.639 mm, respectively. This depicts an accuracy improvement of 51.17 %, reflecting the functioning of the model-based robot stiffness optimization approach.

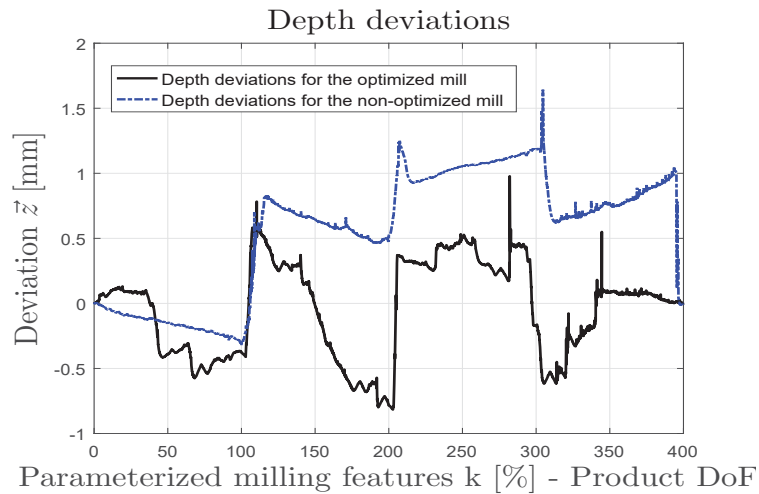


Figure 5.14: Depth deviations for the optimized and non-optimized mills with the single problem strategy.

Experimentation with the multiple problem strategy

The workpiece localization for this milling experimentation is performed using a laser tracker system AT-901. A coordinate system is measured in the robot TCP. Another coordinate system is measured in the workpiece corner, where the reference frame in the CAD model of the workpiece is located. Both coordinate systems are determined by measuring geometries and robot movements. By computing the kinematic chain, the workpiece localization in the real scene is calculated by having as input the robot pose in which the TCP was measured. The same procedure was used for localizing both milled squares.

The experimental machining was done with automatic generated programs in SE^3 based on the paths computed and depicted in Figure 5.11 in page 130. The same CMM and procedure for measuring the milling results are used as in the single problem strategy. The non-optimized path was generated and milled using the same workpiece localization method and with the same conditions (tool wear, environmental condition, robot conditions).

As with the single problem strategy, the parallelism of the sides of the optimized mill are not better than the ones with the non-optimized path. Angle deviations between the parallel sides for the non-optimized and optimized square are -0.024 Deg and -0.032 Deg, respectively. For the perpendicular sides, the angle deviations are -0.027 Deg and -0.037 Deg, respectively. The reason for this is the robot calibration. Figure 5.15 depicts the machined workpieces with and without optimization, focusing on the representative depth deviations. The angle deviations for the optimized mills with the multiple problem strategy are less than with the single strategy due to the avoidance of discontinuities. Due to this avoidance, the robot does not change its orientation drastically during the milling.

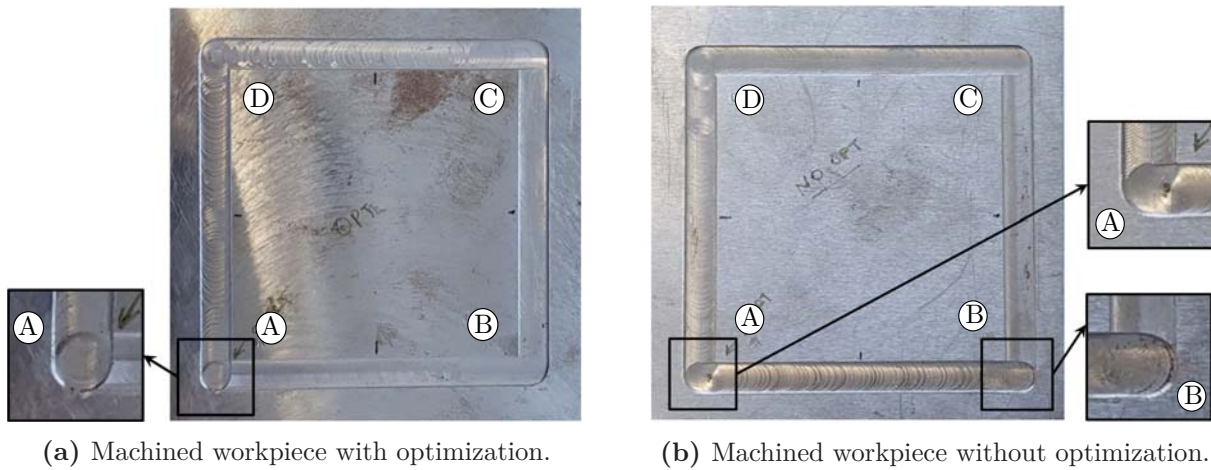


Figure 5.15: Machined workpieces with corners A, B, C, and D using the multiple problem strategy with and without optimization.

Figure 5.16 depicts the depth deviations for the milling process using the multiple problem strategy. Improvement in the milling depth is also noticed in this experimentation. The error for the first depth with respect to the programmed milling depth (1 mm) are -0.024 mm and -0.15 mm for the optimized and non-optimized path, respectively. As for the previous strategy, the zero deviation is defined for the first depth measurement. The mean absolute deviation of the milling depth was computed for the optimized and non-optimized paths as 0.092 mm and 0.216 mm respectively. This

depicts an accuracy improvement of 57.40%, reflecting the improvement of the model-based robot stiffness optimization approach.

The mean absolute deviation is better using this strategy than the single one, this is attributed due to the avoidance of discontinuities in the RMP C-space and the used workpiece localization method. A laser tracker offers better measurement accuracies than the 2D linear scanner used in the previously presented results. The pick observed in the optimized mill in Figure 5.16 corresponds to the depth difference between the start and goal. For the non-optimized mill, picks are measured when starting the second linear feature and when finishing the milling. These picks are correlated with the focused images in Figure 5.15.

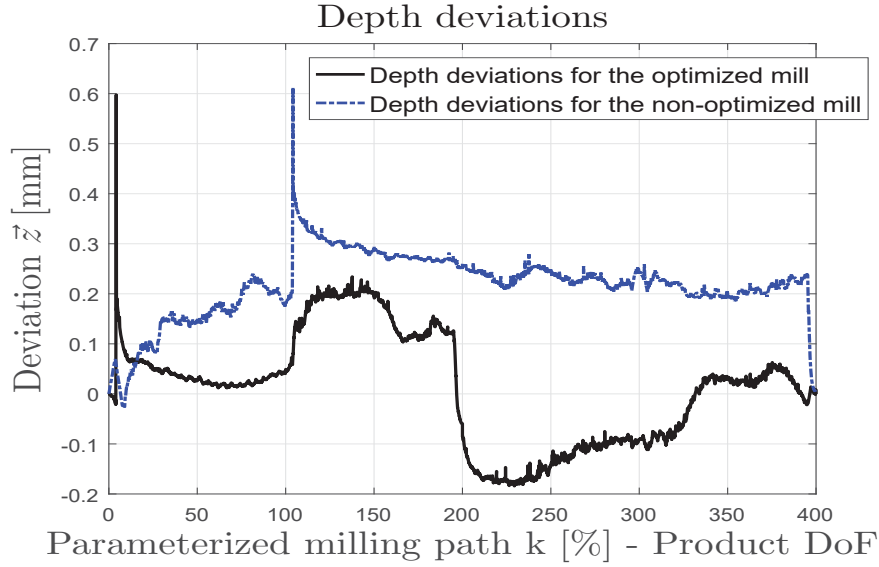


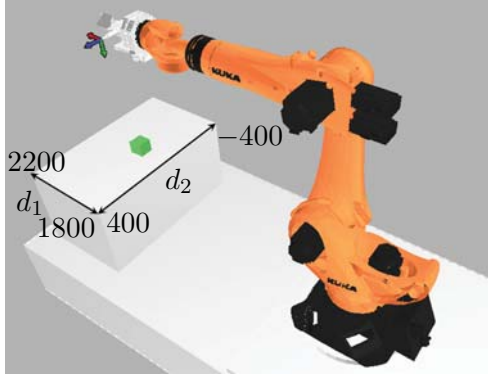
Figure 5.16: Depth deviations for the optimized and non-optimized mills with the multiple problem strategy.

5.5.4 Close-to-Optimal Workpiece Positioning for Robotic Milling

The automatic workpiece positioning has been also implemented for the robotic milling process with a static table in which the workpiece can be translated (Diaz Posada, Mukherjee et al. 2018). The translation of the workpiece is defined with two degrees of freedom denoted as d_1 and d_2 as depicted in Figure 5.17a with their bounds. Fig 5.17b depicts the real robot cell.

The stiffness optimization state cost (here ${}^m\mathbf{C}_s$) has been presented in Section 5.3 and it is used on the sample-based optimal path planning for improving the machining quality. In contrast to the presented implementation in Subsection 5.5.2, the deviations are computed in μm in order to make changes to the workpiece positioning optimization algorithm more visible. The functional DoF of the milling process is defined as $\delta\gamma$, being the rotation around the tool axis as in the previous subsections. A circular product model as defined in Equation (3.3) in page 53 is selected in this use-case. The circular feature selected over the STL format of the workpiece is depicted in Figure 5.6 in page 126.

The total motion cost for the milling use-case for optimizing the workpiece position (${}^m\mathbf{C}_t$) is defined as the integral state cost over the parameterized k dimension of the product feature with backward policy $\mathbf{C}_p$. Note that in this close-to-optimal optimization, the used cost is an integral one and not



(a) Simulation of the milling robot, static table, and workpiece. (b) Real milling robot cell. Photo (Fraunhofer IPA, Rainer Bez)

Figure 5.17: Simulation and real robotic milling cell.

the mechanical work as in Subsection 5.5.2 (refer to Equation 5.15). The reason for this is that the algorithm used for optimizing the path is in this use-case an RRT* and not a T-RRT. Due to the nature of the T-RRT, the optimization is finished once a path is found. Instead, the RRT* allows further exploration of the $\mathbb{RMP}^2$ for improving the path performance under a certain specified time. The further improvement after finding a path is more suitable for optimizing the workpiece position and therefore the choice of the RRT* algorithm. The backward motion cost C_p is also added here. The following equation describes the total motion cost,

$${}^mC_t(\sigma) = \int {}^mC_s dk + C_p. \quad (5.15)$$

Figure 5.18 depicts the costmap of the optimized stiffness for different workpiece positions. The costmap was generated on iterations each 20 mm. The planner was allowed to optimize for 60 seconds. As defined in Section 3.5, a simulated annealing algorithm run for computing a close-to-optimal workpiece position. The orange rectangle symbolizes the start of the simulated annealing and the orange hexagonal star, the end of the algorithm. The results of two simulated annealings (T_w^{wp} when mapped to $\mathbb{RMP}^2$) are depicted showing that the algorithm converges to close-to-optimum values of the total motion cost.

Figure 5.19 illustrates the planning in a specific workpiece pose in which the state costs are less than in the ones of Figure 5.20. Note that in the path planning depicted in Figure 5.19 the minimum of the scale is of approximate $900 \mu m$ while in the planning depicted in Figure 5.20 is of approximate $2100 \mu m$. Note that in the $\mathbb{RMP}^2$, the k dimension defines the constrained circular path for the milling use-case. Due to the nature of the process, the DoF is complete and therefore values close to 180 Deg are allowed to interconnect with values close to -180 Deg as in Figure 5.19. The optimized path in $\mathbb{RMP}^2$ (σ^*) is depicted with black bubbles and lines.

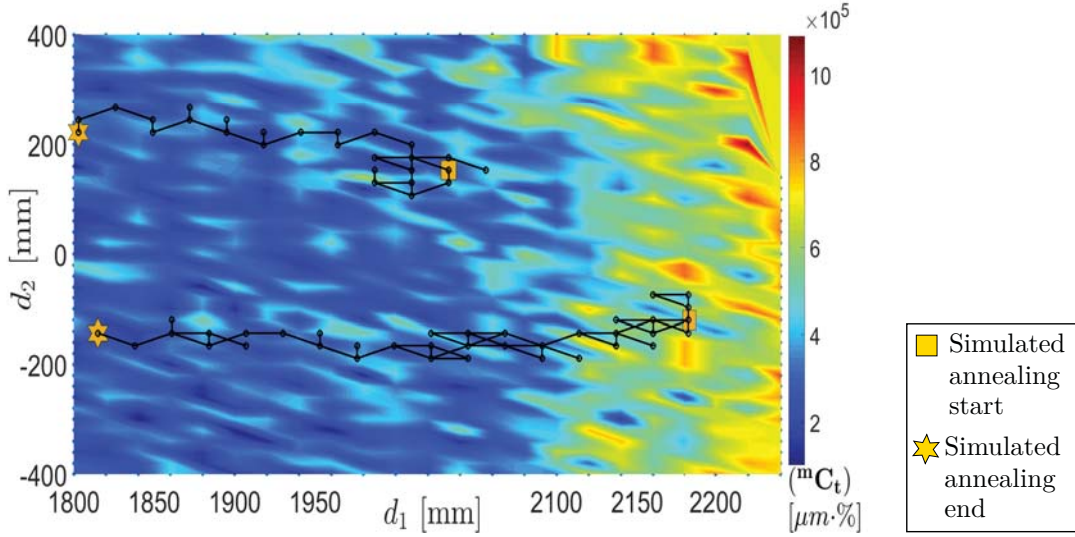


Figure 5.18: Total motion cost mC_t for different workpiece positions with the results of two simulated annealings.

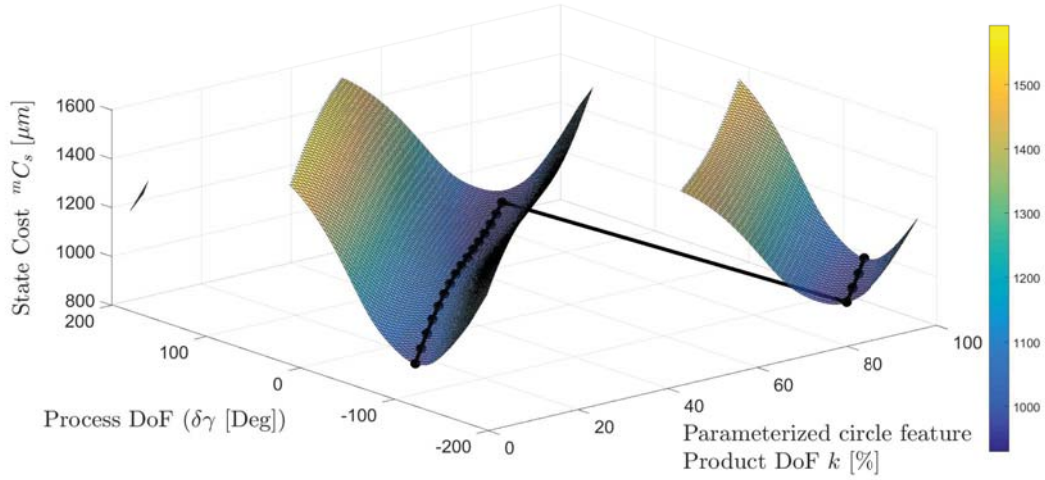


Figure 5.19: Path planning (black bubbles and lines) in the costmap for optimizing translation deviations due to robot compliance in the optimized workpiece position $d_1 = 1800$ and $d_2 = 200$.

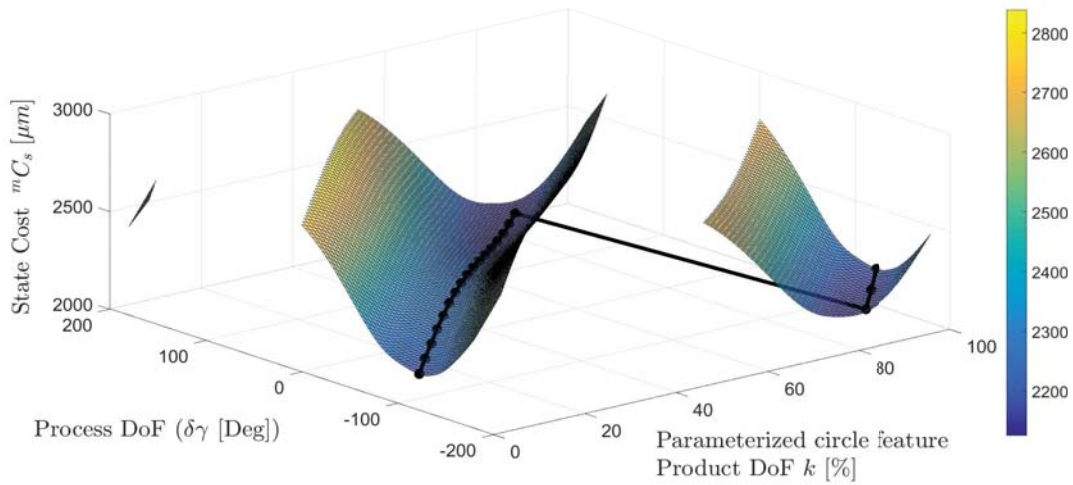


Figure 5.20: Path planning (black bubbles and lines) in the costmap for optimizing translation deviations due to robot compliance positioning the workpiece in the worst position ($d_1 = 2200$ and $d_2 = 400$).

5.6 Conclusion

This chapter described the use-case of optimized stiffness for robotic milling. The improvement of the milling depth accuracy due to the path optimization computed in the general proposed approach and architecture is the main contribution of this use-case. The state cost has been defined as the translation deviations as a result of robot compliance caused by wrenches applied in each joint due to process forces and joint weights. The total motion cost has been defined as the addition of the mechanical work motion cost for the defined state cost and the backward motion policy. The simulated and milled product model consists of four linear features forming a square. Two different strategies used for the interpretation of the product and the definition of the path planning problem have been addressed. Focus on these two strategies has been given in this use-case due to the presented discontinuities in the costmap. The discontinuities are a result of the nature of the state cost with respect to the product feature definitions.

A T-RRT sample-based algorithm has been used for planning using the single problem and multiple problem strategy definition. In the single problem strategy, the discontinuities are not avoided in $\mathbb{RMP}^2$ and therefore significant variation in the robot orientations are obtained in the final robot program in $\mathbb{SE}^3$. This optimized path has been compared with a robot path following the product conventions depicting an improvement with the optimization of 51.17 % with respect to the mean absolute deviation of the milling depth. The optimized path with the multiple problem strategy, has been compared with a path which does not change its orientation in $\mathbb{SE}^3$. By means of this strategy, the discontinuities in the $\mathbb{RMP}^2$ in between product features are avoided. Improvement of the mean absolute deviation of the milling depths are of 57.40 % with the multiple problem strategy. Improvements in the x - y plane of the millings are not observed due to the robot and end-effector inaccuracies for both strategies, recalling the efforts of further research and improvements in this regard. As an extension to the approach for this use-case, the close-to-optimal workpiece positioning for robotic milling is also simulated.

6 Optimized Sensing Strategy for Robotic Deburring

In this chapter, the use-case of optimized sensing strategy for robotic deburring as defined in Subsection 2.7.2 is described. The main difference of this use-case in comparison with the others is the involvement of a sensing system. An optimal definition of the measurement with the sensor is introduced for defining the costmap for automatically programming a sensor-guided deburring process. Moreover, this process has as particularity the requirement of a state machine, an issue that is also addressed using the proposed approach and architecture by setting additional algorithms which facilitate the automatic programming of the state machine.

This chapter is structured as follows: Section 6.1 formally defines the problem. Section 6.2 describes the PPR model of this use-case. Section 6.3 introduces the state and motion cost. Section 6.4 describes the interpretation and planning of the optimized sensing strategy for robotic deburring. Section 6.5 presents the simulation of the optimized paths. Section 6.6 concludes this use-case.

6.1 Problem Definition

For improving the robotic programmability of a sensor-guided robotic process, the following path generation problem is defined: compute automatically and offline a feasible and optimized path σ^* optimizing the sensor measurements by using the DoF of the deburring process $\delta\gamma$, while at the same time constraining the robotic path to the user-specified product features. Figure 6.1 depicts the sensor-guided deburring process in which a 2D measurement of the workpiece is taken before the deburring in order to compensate for workpiece and robot deviations in a later stage of the machining process.



Figure 6.1: Illustration of the sensor-guided process. Photo (Fraunhofer IPA, Rainer Bez)

6.2 Product, Process, and Resource Modeling

This section describes the proposed PPR model representation for this use-case.

6.2.1 Product and Process Models

The product model for this use-case consists of five linear features ($u = 5$) intended to be deburred with sensor compensation. Each linear feature is modeled as described in Subsection 3.1.1 under the multiple problem strategy ($\rho = u$) for describing the transformation world to product $T_w^p(k)$. The selected features are interpreted as defined in Subsection 3.1.1 with multiple planning problems (refer to equation (3.5) in page 55).

Product reference frames are defined starting and finalizing the edge as for the milling case. Orientations of these reference frames are defined with the quaternion q_m of each feature frame (m). Figure 6.2 depicts the STL model of the workpiece in the simulation environment. The coordinate frames depicting the convention defining each feature are displayed also in this figure (machining directions $\vec{x}$ -red vectors- and tool vectors $\vec{z}$ -blue vectors-).

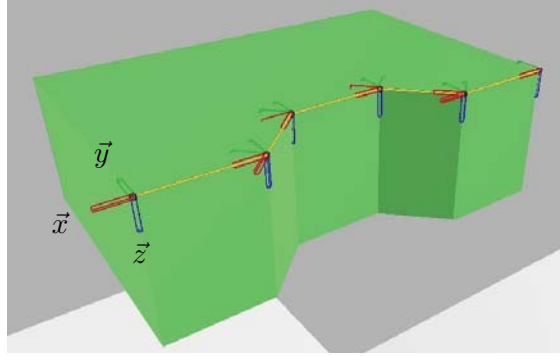


Figure 6.2: Illustration of the product model, the selected linear features, and its coordinates for the optimized sensing strategy use-case.

The process model of the given use-case is analogous to the milling one presented in Subsection 5.2.2. The DoF of the deburring process is also $\delta\gamma$ being the rotation around the tool axis $\vec{z}$ (blue axis in Figure 6.2 following the RGB convention). Due to the geometrical relation between the position of the sensor and the deburring spindle, the bounds of this DoF are defined for the minimum as $\delta\gamma_{min} = -50$ Deg and for the maximum as $\delta\gamma_{max} = 50$ Deg of rotation around the tool axis. The transformation T_p^{proc} is defined as in Equation (5.1).

The kinematic arrangement of the sensor with the deburring tool in the current setup makes the sensor traverse a path ahead of the tool. The kinematic relationship between the tool (tcp) and sensor frame (sen) is used to generate an offset path and a mapping distance d between the sensor measurement frame (mea) and the tool frame. This distance is calculated with the equations which constrain the first product feature to be deburred and then numerically evaluated by giving the known information of the transformations: robot base (rb) to TCP (T_{rb}^{tcp}), TCP to sensor (T_{tcp}^{sen}), and sensor to expected measurement of the workpiece (T_{sen}^{mea}) (as detailed explained in (Diaz Posada, Kumar et al. 2016)). These transformations are depicted in Figure 6.3.

Later, the entire path is modeled into four process states (sensing, deburring, stop sensing, and stop deburring) as shown in Figure 6.3. A state machine model ensures a reliable system by restricting the system to be at only one of the four states at any point of time. The individual states are as follows: (1) Sensing: the tool is not in contact with the product. The robotic system is ready to receive sensor measurements and compute the tool position. (2) Deburring: the tool is in contact with the product. The compensation is active and simultaneously sensor measurements are recorded. (3) Stop sensing: the tool is in contact with the product. The sensor measurements are not recorded but the compensation mechanism is active. (4) Stop deburring: the tool is not in contact with the product and the sensor system is also in a deactivated state.

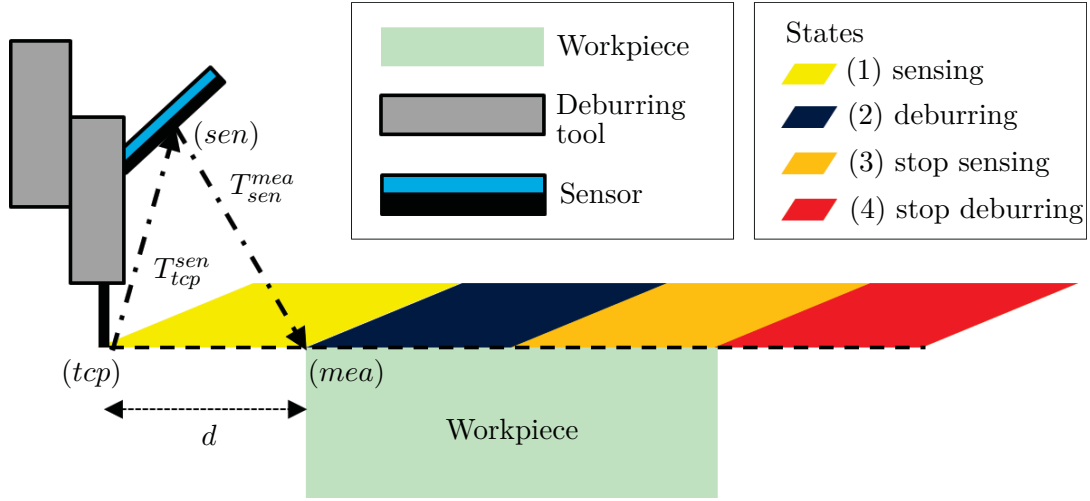


Figure 6.3: Deburring process state machine. (1) sensing, (2) deburring, (3) stop sensing, and (4) stop deburring.

6.2.2 Robot and Sensor models

The resource model (ROBOT), being a 6 joint rotatory articulated robot, is defined with the forward and inverse kinematics as defined in Subsection 3.1.3. An end-effector is mounted to the robot TCP with a deburring tool and a 2D sensor as depicted in Figure 6.1. The transformations flange to TCP and flange to sensor have been calibrated as described in (Diaz Posada, Kumar et al. 2016).

The sensor model is implemented in the simulation environment. The sensor field of view is modeled as a trapezoid taking into account the specifications of the sensor as depicted in Figure 6.4. The collision detection algorithm (refer to Subsection 3.2.1) has been further programmed in order to retrieve the intersection points between the workpiece and the sensor field of view in the simulation. Based on these intersection points, vectors describing the workpiece surfaces are computed as depicted in Figures 6.4 and 6.5 ($\vec{V}_1$ and $\vec{V}_2$).

The intersection point (P_i in Figures 6.4 and 6.5) between these two vectors is also computed and defined as the translation component of the expected measurement in the workpiece. The deburring direction $\vec{x}$ is defined as the vector described by two consecutive translation components (P_{i-1} and P_i). Using the deburring direction and one of the vectors describing a product surface, the expected measurement coordinate frame is computed (mea). Moreover, the sensor model has been programmed

in order to determine if the sensor field of view is not intersecting the workpiece, constituting this the validity function ($\text{IsValid}(\cdot)$) which defines the feasible and infeasible regions of the $\mathbb{RMP}^n$.

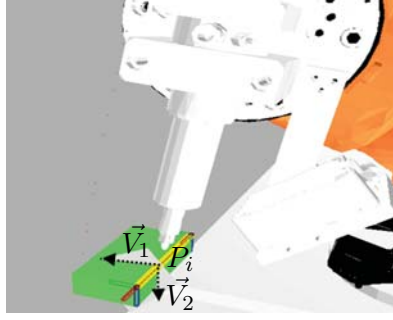


Figure 6.4: Sensor measurement model for computing the expected measurement calculation.

6.3 State and Motion Cost

6.3.1 State Cost

In order to define the state cost of a measurement, two criteria have been taken into account based on the empirical evaluation performed with the sensor and the algorithm which interprets the real 2D profile. The empirical evaluation determined that the algorithm works more robustly and accurately if information of both measurement features of the 2D profile is available. In order to codify this criterion, a symmetry cost has been defined as the relation of the magnitude of the vectors $\vec{V}_1$ and $\vec{V}_2$ as depicted in Figure 6.5 and expressed as follows,

$$\mathbf{C}_{\text{sym}}(k, \delta\gamma) = \frac{\|\vec{V}_1\|}{\|\vec{V}_2\|} \quad (6.1)$$

As second criterion, and as specified in the sensor provider specification, it is desired that the measurement is taken in a specific area of the trapezoid. In order to decode this, a depth cost is also established in order to condition the place in which the 2D profile is taken. This criterion is established as the relation of the magnitude of $\vec{V}_3$ (refer to Figure 6.5) and the desired depth d_{des} as follows,

$$\mathbf{C}_d(k, \delta\gamma) = \frac{\|\vec{V}_3\|}{d_{des}} \quad (6.2)$$

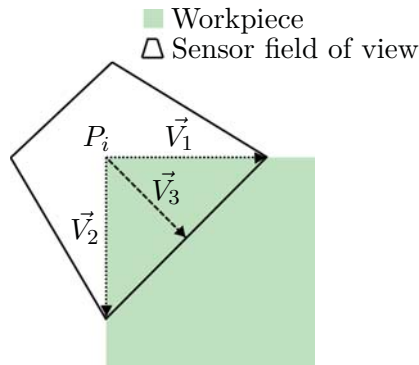


Figure 6.5: Illustration of the state cost for the deburring use-case.

Finally, the state cost for the deburring use-case is defined based on the two previously introduced criteria as denoted with the following function,

$$\mathbf{C}_s(k, \delta\gamma) = \left(1 - \left(\frac{\mathbf{C}_{\text{sym}} + \mathbf{C}_d}{2}\right)\right)^2 \quad (6.3)$$

Note that if both costs ($\mathbf{C}_{\text{sym}}$ and $\mathbf{C}_d$) are equal to one, meaning that the magnitude of $\vec{V}_1$ and $\vec{V}_2$ are equal and that the depth of the measurement $\vec{V}_3$ is as desired, the state cost is zero representing the most optimal measurement as depicted in Figure 6.5. Deviations from this optimum are charged as in Equation (6.3). The cost is computed in the $\mathbb{RMP}^n$ by giving the sampled dimensions k and $\delta\gamma$, transforming these to $\mathbb{SE}^3$ for finally performing the computation in the simulation environment.

6.3.2 Motion Cost

The manufacturing motion cost for the deburring process is defined as the integral function of the cost of the interconnected states through all the path (σ). Similar to the welding and milling use-cases, the backward policy cost is added in case that backward states in the k dimension in the $\mathbb{RMP}^n$ are interconnected ($\mathbf{C}_p = \infty$ for backward motions). The following equation notates the motion cost for the deburring process,

$$\mathbf{C}_t(\sigma) = \int \mathbf{C}_s dk + \mathbf{C}_p. \quad (6.4)$$

6.4 Optimized Path Generation with Sensing Strategy Optimization

The sensing strategy optimization is based on the general proposed methodology and architecture (refer to Chapter 3). The PPR model is interpreted as in the other use-cases. The main difference of this use-case is the calculation of the state and motion cost based on a sensor model. Moreover, the interpreted paths to be deburred are offset with the computed parameter d which is part of the product model. The general algorithm defined in Section 3.4 is approached for the computation of the paths optimizing the sensing as described in the following algorithm,

Algorithm 9 Automatic interpretation and computation of the sensing strategy optimization for robotic deburring using an RRT* algorithm.

Input: P (Equation (3.5)), PROC (Equation (5.1) and d), ROBOT (as defined in Subsection 5.2.3), $\mathbf{C}_t$ (equation 6.4), IsValid (as defined in Subsection 6.2.2), ρ

- 1: **for** $m = 1$ to ρ **do**
 - 2: $T_w^{proc}(k, \delta\gamma) = {}^m\mathbf{f} \leftarrow \text{RMPtoSE}^3({}^m\mathbf{P}, \text{PROC}, \text{ROBOT}, d)$
 - 3: $[{}^m s_s, {}^m s_g] \leftarrow \text{StartGoalStates}({}^m\mathbf{P}, \text{PROC}, \text{ROBOT}, \mathbf{C}_s, \text{IsValid}, d)$
 - 4: ${}^m\mathbf{B} \leftarrow \text{Bounds}({}^m\mathbf{P}, \text{PROC}, d)$ as in Equation (5.12) or (5.13)
 - 5: ${}^m n \leftarrow \text{DetermineDim}({}^m\mathbf{P}, \text{PROC})$
 - 6: {Sample and calculate optimized path for each feature}
 $\sigma \in [{}^m k_s, {}^m k_g] \leftarrow \text{RRT}^*(\mathbb{RMP}^2({}^m\mathbf{f}, n, {}^m\mathbf{B}), {}^m s_s, {}^m s_g, \mathbf{C}_t, \text{IsValid})$
 - 7: $m = m + 1$
 - 8: **end for**
 - 9: **return** $\sigma^* \in [{}^1 k_s, {}^\rho k_g]$ {retrieve concatenated optimized path}
-

6.5 Optimization in OLP Software

To evaluate the proposed sensor-guided robotic deburring use-case, the previously described models, and state and motion costs have been tested in simulation as described below.

6.5.1 Experimental Setup

The architecture of the system is designed to reduce programming and set-up times by modeling the task and the process model. The robotic deburring process is semantically modeled into a software using the AML and the XML formats as for the other use-cases. Based on the task description, the robotic deburring process is interpreted by the path generation software component to generate deburring paths constraining the motions.

The software architecture of the robotic system facilitates the visualization of the robot cell and the interaction with the workpiece. The robotic deburring system is depicted in Figure 6.6. The cell consists of a Kuka Quantec KR270 R2700 ultra industrial robot manipulator, a Biax pneumatic spindle deburring tool with adjustable deflection and a 2D laser scanner, scanControl 2900-50 from the company Micro-Epsilon mounted on the robot flange. The spindle allows the configuration of different stiffnesses and deflections providing the system with a range of tolerance when deburring the workpiece. The laser scanner provides the robotic system with measurements of the workpiece to compensate the robot using a control system after the optimized program generation (Diaz Posada, Kumar et al. 2016).

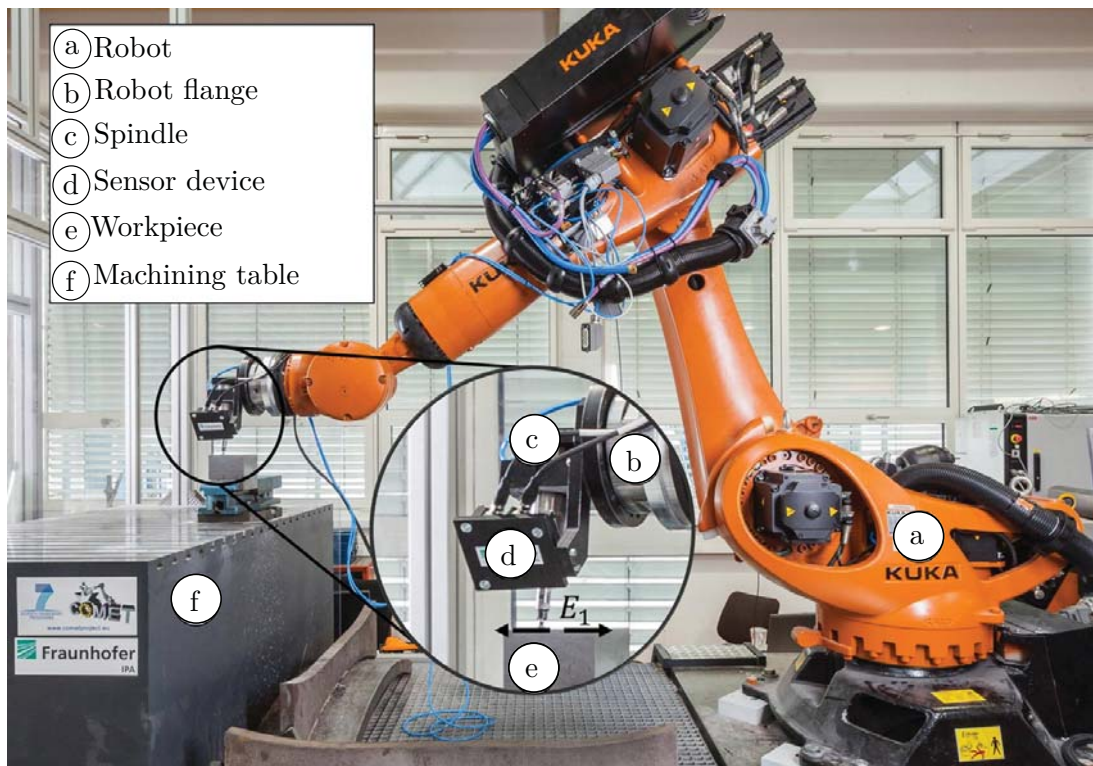


Figure 6.6: Deburring robot cell. Original photo (Fraunhofer IPA, Rainer Bez)

6.5.2 Optimization in OLP Software

As depicted in Figure 6.2 in page 140, a five edges workpiece has been loaded into the OLP software as an STL file in order to allow the end-user to select the edges. The deburring process model is linked from the AML file and the algorithm for computing the offset d is also referenced from this product model. The overall architecture is then used in order to interpret the $\mathbb{RMP}^2$ for later sampling and planning over it using an RRT* algorithm for each feature.

Figure 6.7 depicts the state cost elevation map with the computed path over it and the infeasible states from a 3D perspective after allowing planning for 120 s using an RRT* algorithm and interpolating the final path for each feature in 20 discretizations. Infeasible states are robot configurations where the sensor cannot measure the workpiece (transparent spaces in the figure).

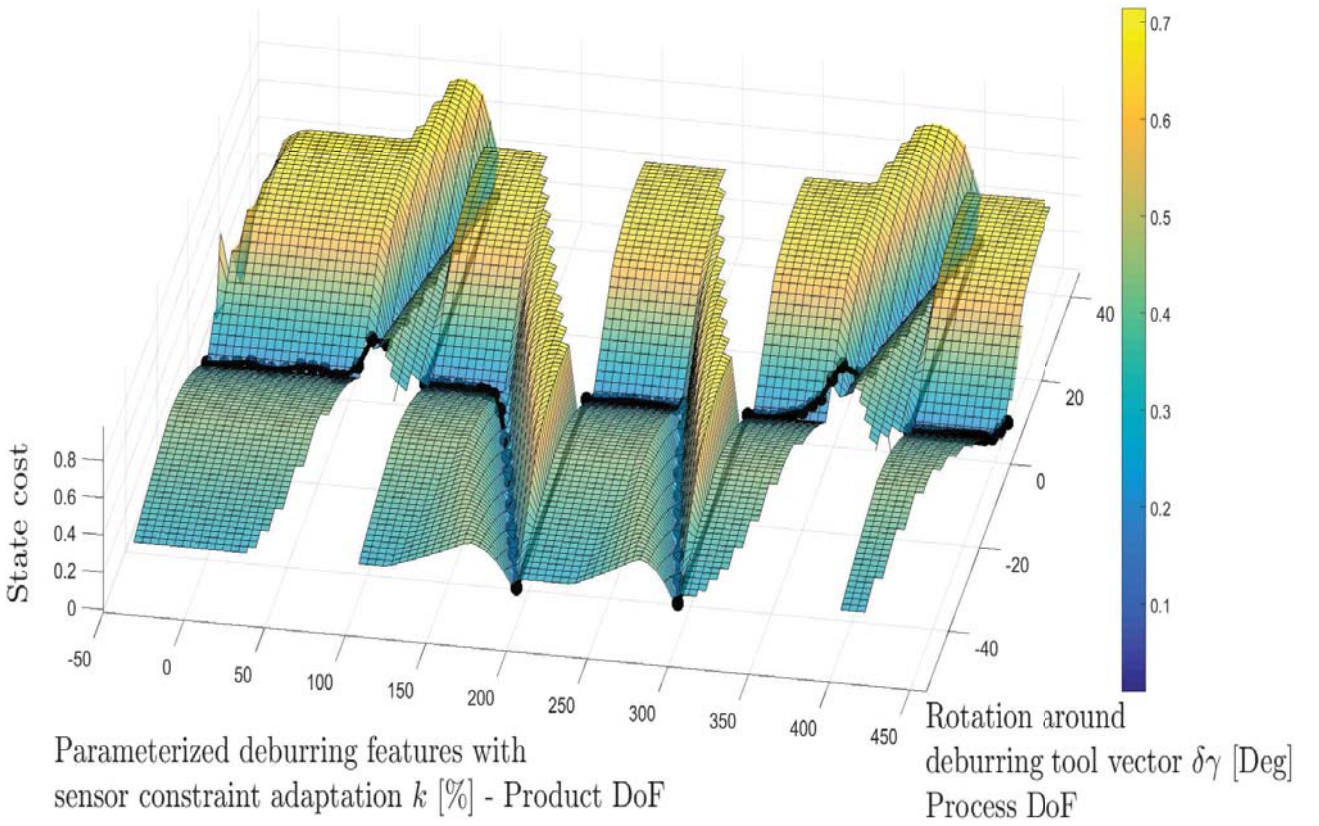


Figure 6.7: Sensing state cost elevation map for the deburring use-case and final computed optimized path (black bubbles and lines) for the five product edges workpiece.

The $\mathbb{RMP}^2$ depicted in Figure 6.7 does not start in zero in the k dimension in comparison with the other use-cases (refer to Subsections 4.5.2 and 5.5.2). The reason for this is the state machine implemented and the computed parameter d (refer to Subsection 6.2.1) which offset the path. For this use-case, the multiple problem strategy (refer to Section 3.1.1.2) has been used. With this strategy each edge is treated as a different path optimization problem. Figure 6.8 depicts the same planning and $\mathbb{RMP}^2$ with its two dimensions in a 2D view. The infeasible states are also shown in this figure as transparent spaces.

Proper selection of the end and start goals has been done as defined in Section 3.4 for avoiding representatives changes in orientation in the Cartesian space $\mathbb{SE}^3$ when deburring from one to another

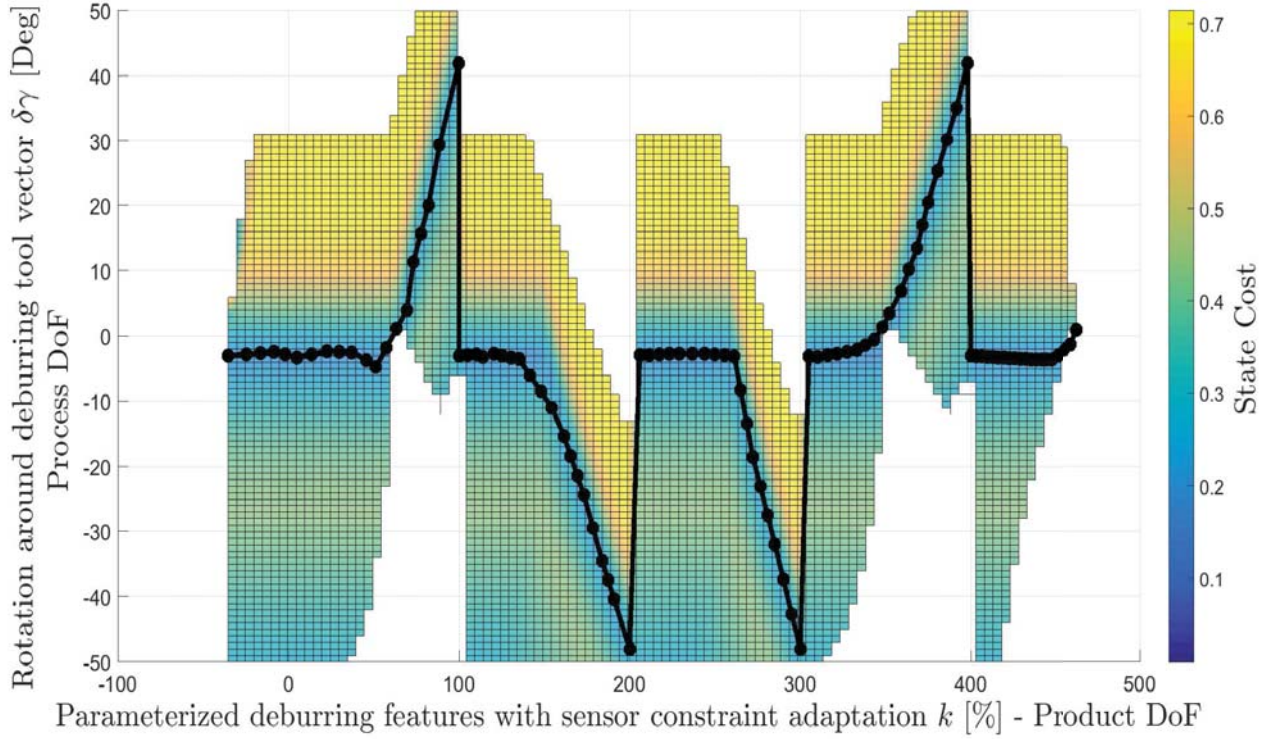


Figure 6.8: $\mathbb{RMP}^2$ for the five product edges with its sensing state costs and the computed optimized path (black bubbles and lines) using the RRT* algorithm for the deburring use-case.

edge. In the final computed path abrupt changes in orientations of the process DoF are present in $\mathbb{RMP}^2$ due to this selection. These abrupt changes are due to the discontinuities of the state cost values between different edges as can be seen in Figure 6.7. It is worth emphasizing that these abrupt changes in $\mathbb{RMP}^2$ are not abrupt changes in the Cartesian space due to the implemented heuristic constraining the start and goal states of following features.

The computed optimized path in $\mathbb{SE}^3$ in several steps over the workpiece outlining the sensor-guided process is shown in Figure 6.9. In steps 2, 4, 6, and 8, the change of orientation with respect to the product orientation definition optimizing the measurement can be observed. The start position of the robot accomplishing the sensor constraints and the complete path is depicted in Figure 6.10.

Results and characteristics of the path planning specifying the different maximum allowed sampling times using the RRT* algorithm for the sensor-guided deburring use-case are listed for all product features in Table 6.1. From the table, the asymptotic behavior of the optimization can be noticed with the improvement of final cost when planning longer times for almost all features. Exceptions are due to the sample-based nature of the used algorithms.

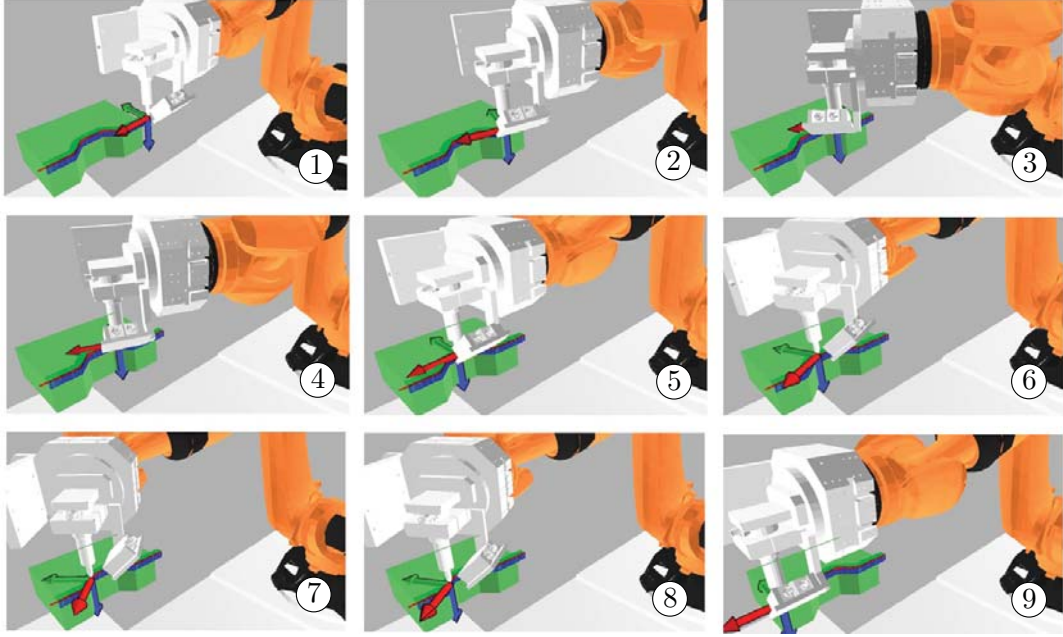
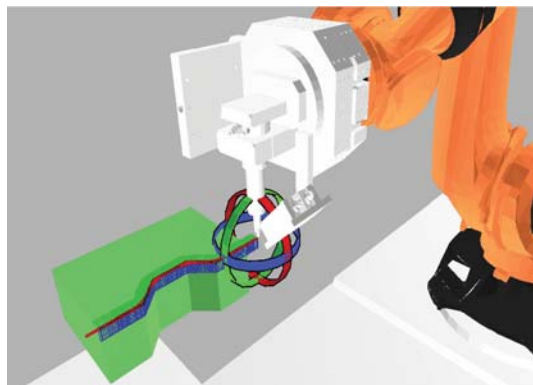


Figure 6.9: Sequence of robot motions for the sensor strategy optimization.

Table 6.1: Times, results, and characteristics of the path planning for different maximum allowed sampling times using the RRT* algorithm for the sensor-guided deburring use-case.

Allowed sampling time [s]	Product feature Nr.	Initial total cost	Final total cost	Evaluated states Nr.	Rewires Nr.	Interpretation time [s]	Plotting of space time [s]	Transformations time [s]	Total time [s]
10	1	37.59	37.587	29	320	10.25	2.374	7,353	69.977
10	2	37.22	37.217	32	368				
10	3	31.55	28.603	31	352				
10	4	35.38	33.981	41	519				
10	5	15.06	14.163	37	451				
30	1	73.30	33.523	116	2018	9.446	2.131	6.987	168.564
30	2	31.98	31.174	117	2040				
30	3	39.10	32.050	126	2238				
30	4	38.50	28.273	156	2921				
30	5	15.06	13.598	154	2875				
120	1	35.69	30.242	808	21049	9.306	2.038	7.791	619.135
120	2	35.61	25.797	1016	27511				
120	3	39.10	23.939	986	26567				
120	4	54.11	24.851	1025	27799				
120	5	17.43	13.742	973	26164				

Figure 6.10: Computed optimized path in $\mathbb{SE}^3$ over the deburring workpiece.

6.6 Conclusion

This chapter described the use-case of optimized sensing strategy for robotic deburring. The manual programming of this use-case is time-consuming due to the complexity of accomplishing the deburring constraints while rotating the robot TCP for getting optimized or valid measurements with the sensor. The optimization of the sensing strategy in the general approach demonstrated in simulation is the main contribution of this use-case. The state cost has been defined with two criteria on the empirical evaluation performed to a 2D laser scanner. A symmetry cost is defined for getting information at its maximum of both measured features and a depth cost is introduced in order to condition the place in which the 2D profile is taken. The state is defined as feasible if the field of view of the sensor is intersecting the workpiece for taking a measurement. Moreover, a state machine has been implemented for taking into account the different steps of the sensor-guided process. The optimization has been simulated for a product with five linear continuous features using an RRT* sample-based algorithm and a path problem automatically interpreted, defined, and configured using the proposed approach and architecture.

7 Conclusions and Outlook

7.1 Conclusions

This thesis presents a novel approach for interpreting methodologically the components of the robotic manufacturing process using as inputs the proposed product, process, and resource classification and models for automatically generating offline optimized robotic paths, by procedurally configuring and deploying sample-based algorithms. The approach and architecture are based upon and propose the synergy between the following four main robotic research areas: (1) the offline programming for robotic manufacturing processes, (2) the sample-based path generation, (3) the knowledge representation in robotic manufacturing and (4) the specific modeling of the robotic manufacturing processes studied as use-cases. This approach is proposed to make up for the deficiency that the path generation for robotic manufacturing processes is time-consuming and complex, expert-dependent, and its optimization is primarily manual.

The approach is based on the structured interpretation of the proposed model-based description of the robotic manufacturing process. Furthermore, a novel robotic manufacturing process configuration space is introduced. The Cartesian translational product constraints and the optimal or reference orientations of the product model are mapped in one dimension of this configuration space. The rotational process degrees of freedom are represented as further dimensions. The product, process, resource, and end-user defined task specification are taken into account for automatically generating the required transformations for mapping between the proposed configuration space and the Cartesian and Joint space. Moreover, an automatic termination heuristic is defined in order to determine when a solution is optimized by taking into account the asymptotic nature of the used algorithms. Based on the interpretation, a path planning problem and a sample-based algorithm are automatically configured for allowing computation of paths, optimizing the defined process and motion criteria while avoiding infeasibilities such as collisions, unreachabilities, and/or maximum joint limits.

The architecture is implemented using inputs, which are further developed, from widely used models, semantic descriptions, and software packages. An offline robotic simulation environment has been improved for interpreting the end-user task specification and for executing the proposed approach. The outputs of the approach are optimized paths and ready-to-use robot programs. Two different strategies have been implemented and evaluated for interpreting multiple product features and for defining the path planning problem. The single problem strategy defines one path planning problem for all selected features. The multiple problem strategy determines a path planning problem per feature. This strategy allows the avoidance of discontinuities in the costmaps of the robotic manufacturing processes. The approach is evaluated in three different use-cases that demonstrate its generality and model-based structure.

The first use-case deals with the optimized collision avoidance for robotic welding. Different product features, such as the linear and semi-circumferential weld seams are evaluated. Optimal welding gun orientations are defined for obtaining good welding quality and deviations from this optimum are charged accordingly into the state cost definition. Motion cost policies are used in order to avoid backward robot motions. Furthermore, in this use-case, the constraining of the angular velocity and the planning using multiple process degrees of freedom are presented. Experimentation in a real robot cell demonstrates that the proposed approach, named Automatic Optimized Offline Programming (AOLP), is 1.24 times faster to be explained to programmers and that the programming time is 4.23 times faster than the teach-in approach on average for the given welding task. Moreover, it has been demonstrated that the AOLP improves the homogeneity of the welded throat thickness by 35.62 % in comparison to the teach-in approach for the evaluated use-case. Experimental evaluation of the required programming time using the AOLP in comparison with a commercial offline programming software was also measured as 3.84 times faster on average for the welding use-case.

The second use-case demonstrates the optimized stiffness path generation for robotic milling. Stiffer robot poses, which results in better process quality, are determined with the approach. The milling process intrinsic redundancy is solved with the proposed approach optimizing the Euclidean robot translation deviation, taking into account the compliance model in joint space of an industrial robot. Due to the modular-based structure of the approach, some models are reused from the welding use-case such as the linear features of the product and the backward motion policy. The two different product modeling strategies have been experimented in a real robot-milling of a steel workpiece. Improvements in the depth deviations with the single and multiple problem strategies are calculated as 51.17 % and 57.40 % respectively.

For the milling and welding use-cases, the workpiece close-to-optimal positioning has also been simulated based on an extension of the proposed approach. Evaluation in the welding robot cell of the workpiece positioning has been performed minimizing the robot joint movements.

The third use-case simulates the automatic programming of sensor-guided robotic deburring by optimizing the sensing process while accomplishing the deburring process constraints. The field of view and the symmetry of the profiles are optimized as sensing strategy. Manual programming of this use-case is time-consuming due to the involvement of sensors. Here, empirical criteria are defined for optimizing the field of view of a two dimensional laser scanner and the measurement with respect to the workpiece. The costmap of this manufacturing process has been simulated and the path automatically optimized in reasonable programming times (120 seconds for close-to-optimal solutions) while maintaining the deburring constraints, thus improving the programming experience.

7.2 Outlook

The approach could be further developed in order to take into account robot tool center point velocities within the translational constraints of the product. For achieving this, advances of the kinodynamic planning can be additionally embedded into the proposed architecture.

Moreover, the proposed approach could be linked with high-level planning approaches in order to plan process steps for completely manufacturing products based on the description of the product, process, and resource components. Furthermore, the standardization of models should be defined in order to allow for the information exchange with other systems, such as robot controllers.

Future work based on this approach is the implementation of further models for the product, process, and resource components in other robotic manufacturing processes or use-cases, such as robotic laser-cutting. The extension of product features such as surfaces for robotic painting is also of interest. Moreover, more complex processes models could be linked by relating, for instance, finite-element software.

The proposed approach is complete automatic for the end-user. However, for the implementation of new optimization criterion some parameters of the sample-based algorithms may require initial tuning by the programmers in order to improve their performance. Further work to avoid this tuning is to implement learning techniques for performing this initial tuning automatically. Learning algorithms could be used also to learn the path planning process from the proposed approach to compute the paths faster. Furthermore, informed approaches for sampling taking into account the manufacturing cost could be approach to explore the configuration spaces more effectively.

Appendixes

A. Robot Joint Mass and Center of Mass Estimation

This appendix describes the process for estimating the robot joint masses m_{jn} and center of masses cm_{jn} required in the wrench computation at each joint (refer to Subsection 5.2.3). Moreover, this appendix lists the estimated masses and center of masses for a Kuka KR270 R2700 ultra. The estimation process is based on the CAD model of the industrial robot. From the technical data, the total weight of the robot is known. The complete volume of the robot is estimated using the software SolidWorks. A material with manual set density is assigned to all joints in order to make the CAD model weigh the same as specified in the technical data. After this step, each joint is selected and its mass and center of mass is estimated with the same software. The estimation is done under the assumption that all joints are solid and that they are composed of the same material, which is not true for the real robot. For obtaining the exact center of masses, other methods need to be approached. However, this method was good enough for proving that the stiffness can be optimized in robot motions using sample-based techniques. Figure A.1 depicts the center of masses for each joint. For the last robot joint, the center of mass is shown also with the one of the end-effector. Coordinate systems are depicted using the RGB convention. Table A.1 lists the mass and center of mass for each joint of the Kuka KR270 and its end-effector.

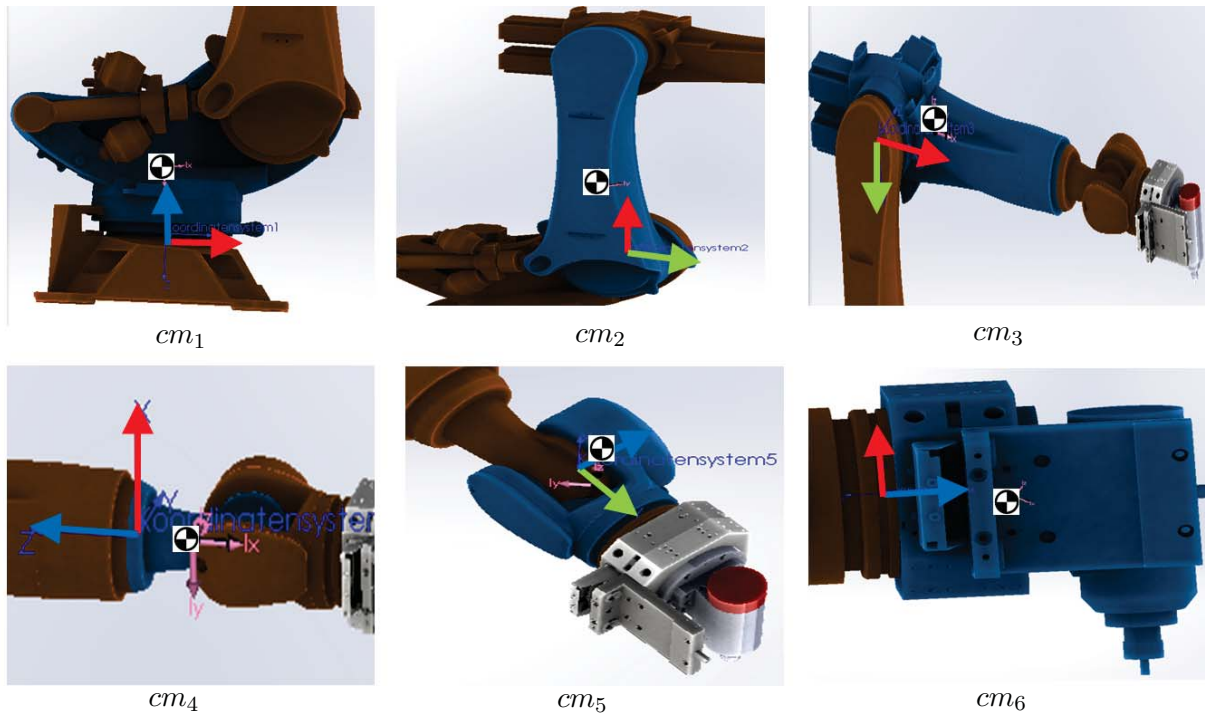


Figure A.1: Illustration of the center of mass for each joint of the Kuka KR270 R2700 ultra.

Table A.1: Mass and center of mass of the end-effector and each joint for a Kuka KR270 R2700 ultra estimated based on its CAD model and its total weight.

Joint	Mass [kg]	cm_x [mm]	cm_y [mm]	cm_z [mm]
Joint 1 (j_1)	395.72	-4.24	-3.06	175.3
Joint 2 (j_2)	239.75	431.54	2.8	-235.73
Joint 3 (j_3)	178.30	222.13	25	3.36
Joint 4 (j_4)	17.60	1.35	0	-154.06
Joint 5 (j_5)	57.93	54.3	0	37.8
Joint 6 (j_6)	3.29	0	0	13.88
End-effector (EE)	40.2	-8.86	12.26	104.41

B. Parameters of the Joint Compliance Model for the Robot Kuka KR270 R2700 Ultra

The values of the joint compliance models ($s_a \dots s_h$) for the six joints of the Kuka KR270 R2700 ultra are provided in Table B.1. Table B.2 describes the lower and upper ranges s_{min} and s_{max} respectively, where the non-linearities are determined for each robot joint. The robot joint compliance parameters have been identified for the Kuka KR270 R2700 ultra using the methodology described by Schneider, Diaz Posada et al. (2015).

Table B.1: Parameters for the linear functions and third order polynomials describing the joint compliance models for the robot Kuka KR270 R2700 ultra.

Joint	s_a	s_b	s_c	s_d
1	5.88×10^{-09}	1.31×10^{-03}	5.71×10^{-09}	-1.41×10^{-03}
2	1.06×10^{-08}	-1.55×10^{-03}	1.23×10^{-08}	1.00×10^{-03}
3	1.37×10^{-08}	9.47×10^{-04}	1.70×10^{-08}	-4.04×10^{-03}
4	6.77×10^{-08}	-6.13×10^{-03}	7.14×10^{-08}	-2.10×10^{-03}
5	4.27×10^{-08}	5.01×10^{-04}	5.92×10^{-08}	2.01×10^{-03}
6	1.74×10^{-07}	-2.64×10^{-03}	1.23×10^{-07}	7.74×10^{-06}

Joint	s_e	s_f	s_g	s_h
1	-1.33×10^{-18}	5.03×10^{-16}	3.13×10^{-08}	-4.68×10^{-05}
2	6.82×10^{-21}	-1.10×10^{-14}	1.23×10^{-08}	1.00×10^{-03}
3	-2.59×10^{-20}	-4.09×10^{-14}	6.32×10^{-09}	5.89×10^{-04}
4	8.04×10^{-18}	-1.84×10^{-14}	9.19×10^{-09}	-4.16×10^{-03}
5	1.40×10^{-19}	-2.24×10^{-14}	4.29×10^{-08}	5.58×10^{-04}
6	6.28×10^{-17}	5.22×10^{-12}	1.74×10^{-07}	-2.64×10^{-03}

Table B.2: Limits for the linear functions of the compliance models for the robot Kuka KR270 R2700 ultra.

Compliance Limits	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
s_{min}	-0.8×10^5	0	-9×10^5	-5×10^4	-1.5×10^5	-0.5×10^5
s_{max}	0.8×10^5	1×10^6	-1×10^5	5×10^4	1×10^5	0

C. Robot Tool Calibration and Accuracy

The robot milling tool depicted in Figure 5.5 in page 125 was calibrated using a laser-tracker AT-901. In order to calibrate the system, the robot flange position was computed by projecting a measured circle by moving axis 6 on the flange plane. The $\vec{z}$ vector of the flange is defined as the normal vector of the circle. The robot is moved in another Cartesian axis in order to have a second component to build an orthogonal frame. The vector of the milling tool is measured with a vector tool, which is mounted on the spindle and which allows measuring two laser-tracker targets. A plane in the spindle is measured in order to compute the tool position by intersecting the measured tool vector and the plane. One of the flange vectors is projected into the spindle plane for constructing another vector, which allows computing an orthogonal frame in the tool. In this way, the transformation from the flange to the tool is calibrated. Moreover, three targets are fixed to the tool in order to measure the position and the rotational accuracy of the tool. Figure C.1 depicts the tool accuracy while rotating around the $\vec{z}$ axis ($\delta\gamma$) from -20 to 30 Deg each 5 Deg, which was the visible range for the laser-tracker. Table C.1 lists the deviations. Deviations of up to 0.35 mm in x , 0.5 mm in y , and 0.47 mm in z are observed for these rotations.

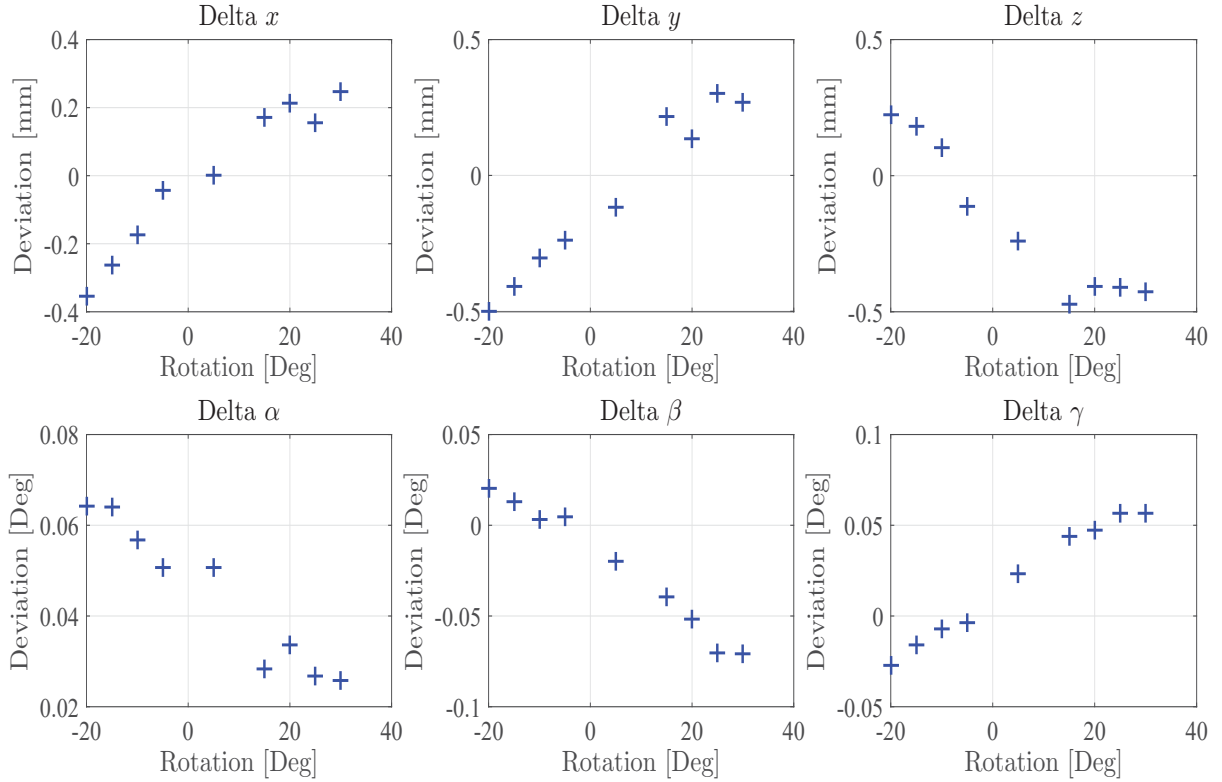


Figure C.1: Robot deviations due to rotation around the z-axis ($\delta\gamma$).

Table C.1: Robot deviations due to rotation around the z-axis ($\delta\gamma$).

Deviation	x [mm]	y [mm]	z [mm]	α [Deg]	β [Deg]	γ [Deg]
Average	0.17	0.27	0.28	0.04	0.03	0.03
Maximum	0.24	0.30	0.22	0.06	0.02	0.05
Minimum	-0.35	-0.50	-0.47	0.02	-0.07	-0.02

Bibliography

- Abele, Weigold et al. (2007) Abele, Eberhard, Weigold, Matthias, and Rothenbücher, Stefan, 2007. Modeling and Identification of an Industrial Robot for Machining Applications, *CIRP Annals - Manufacturing Technology* **Vol. 56**(1) pp. 387–390, doi:10.1016/j.cirp.2007.05.090
- Abele, Bauer et al. (2008) Abele, Eberhard, Bauer, Jörg, Rothenbücher, Stefan, Stelzer, Maximilian, and Stryk, Oskar von, 2008. Prediction of the Tool Displacement by Coupled Models of the Compliant Industrial Robot and the Milling Process, In: *International Conference on Process Machine Interactions*, 3-4 September 2008, Hannover Germany, pp. 223–230
- Abele, Rothenbücher et al. (2008) Abele, Eberhard, Rothenbücher, Stefan, and Weigold, Matthias, 2008. Cartesian compliance model for industrial robots using virtual joints, *Production Engineering* **Vol. 2**(3) pp. 339–343, doi:10.1007/s11740-008-0118-0
- Abele, Bauer et al. (2010) Abele, Eberhard, Bauer, Jörg, Pischán, Matthias, Stryk, Oskar von, Friedmann, Martin, and Hemker, Thomas, 2010. Prediction of the tool displacement for robot milling applications using coupled models of an industrial robot and removal simulation, In: *2nd International Conference on Process Machine Interactions*, 10-11 June 2010, Vancouver Canada
- Abele, Haehn et al. (2016) Abele, Eberhard, Haehn, Felix, Pischán, Matthias, and Herr, Florian, 2016. Time Optimal Path Planning for Industrial Robots Using STL Data Files, In: *5th CIRP Global Web Conference - Research and Innovation for Future Production (CIRPe 2016)*, Patras Greece, 4-6 October 2016, pp. 6–11, doi:10.1016/j.procir.2016.08.038
- Agyapong-Kodua, Haraszkó et al. (2014) Agyapong-Kodua, K., Haraszkó, Csaba, and Németh, István, 2014. Recipe-based Integrated Semantic Product, Process, Resource (PPR) Digital Modelling Methodology, *Procedia CIRP* **Vol. 17** pp. 112–117, doi:10.1016/j.procir.2014.03.118

-
- Ahmed, Yuan et al. (2015) Ahmed, Syeda Mariam, Yuan, Jinqiang, Wu, Yue, Chew, Chee Meng, and Pang, Chee Khiang, 2015. Collision-free path planning for multi-pass robotic welding, In: *IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*, 08-11 September 2015, Luxembourg, pp. 1–4, doi:10.1109/ETFA.2015.7301594
- Alatartsev, Guedemann et al. (2012) Alatartsev, Sergey, Guedemann, Matthias, and Ortmeier, Frank, 2012. Trajectory Description Conception for Industrial Robots, In: *ROBOTIK 2012; 7th German Conference on Robotics*, 21-22 May 2012, Munich Germany, pp. 1–6, ISBN 978-3-8007-3418-4
- Altintas (2000) Altintas, Yusuf, 2000. Modeling approaches and software for predicting the performance of milling operations at MAL-UBC, *Machining Science and Technology* **Vol. 4**(3) pp. 445–478, doi:10.1080/10940340008945718
- Ameri, Akan et al. (2010) Ameri, Afshin E., Akan, Batu, and Çürüklü, Baran, 2010. Augmented Reality Meets Industry: Interactive Robot Programming, In: *SIGRAD 2010 – Content Aggregation and Visualization*, 25-26 November 2010, Västerås Sweden, pp. 55–58
- Arikan and Balkan (2000) Arikan, Sahir and Balkan, Tuna, 2000. Process modeling, simulation, and paint thickness measurement for robotic spray painting, *Journal of Robotic Systems* **Vol. 17**(9) pp. 479–494, doi:10.1016/S0007-8506(07)62124-6
- Arnold (2008) Arnold, Gustavo V., 2008. *Automatization of the code generation for different industrial robots*, Portugal, Minho, Univ., Diss., 2008, URL <http://hdl.handle.net/1822/9167>, retrieved: 21.03.18
- Atmosudiro, Keinert et al. (2014) Atmosudiro, Agus, Keinert, Matthias, Karim, Ali, Lechler, Armin, Verl, Alexander, and Csizar, Akos, 2014. Productivity Increase through Joint Space Path Planning for Robot Machining, In: *European Modelling Symposium*, 21-23 October 2014, Pisa Italy, pp. 257–262, doi:10.1109/EMS.2014.46
- Bauer (2011) Bauer, Jörg, 2011. *Methoden der Offline-Bahnkorrektur für die spanende Bearbeitung mit Industrierobotern*, Berlin: Epubli, ISBN 978-3844215441, Darmstadt, Technischen Univ., Diss., 2011

-
- Bauer, Friedmann et al. (2013) Bauer, Jörg, Friedmann, Martin, Hemker, Thomas, Pisch, Reinl, Abele, Eberhard, and Stryk, Oskar Von, 2013. Analysis of Industrial Robot Structure and Milling Process Interaction for Path Manipulation, In: *Process Machine Interactions: Prediction and Manipulation of Interactions between Manufacturing Processes and Machine Tool Structures*, Berlin Heidelberg: Springer, pp. 245–263, ISBN 978-3-642-32448-2, doi:10.1007/978-3-642-32448-2_11
- Bayan, Pereira et al. (2005) Bayan, Renato Ventura Henriques, Pereira, Carlos Eduardo, Queiroz, Alexandre Bracarense, Guenther, Raul, Dourado, Antonio O., and Martins, Daniel, 2005. Robot Welding Trajectory Planning Using Screw Theory, In: *18th International Congress of Mechanical Engineering*, 6-11 November 2005, Ouro Preto, pp. 196–201
- Behnam, Berbyuk et al. (2010) Behnam, Kamrani, Berbyuk, Viktor, Wäppling, Daniel, Feng, Xiaolong, and Andersson, Hans, 2010. Optimal Usage of Robot Manipulators, In: Jimenez, Agustin and Hadithi, Basil M al (Eds.), *Robot Manipulators Trends and Development*, Intech, ISBN 978-953-307-073-5, doi:10.5772/9198
- Berenson, Srinivasa et al. (2011) Berenson, Dmitry, Srinivasa, Siddhartha, and Kuffner, James, 2011. Task Space Regions: A framework for pose-constrained manipulation planning, *The International Journal of Robotics Research* **Vol. 30**(12) pp. 1435–1460, doi:10.1177/0278364910396389
- Berselli, Pellicciari et al. (2016) Berselli, Giovanni, Pellicciari, Marcello, Bigi, Gabriele, and Andrisano, Angelo O., 2016. Virtual Prototyping of a Compliant Spindle for Robotic Deburring, In: *Proceedings of Second International Conference on Electrical Systems, Technology and Information 2015 (ICESTI 2015)*, Singapore: Springer Singapore, pp. 17–30, ISBN 978-981-287-988-2
- Biagiotti and Melchiorri (2008) Biagiotti, Luigi and Melchiorri, Claudio, 2008. *Trajectory Planning for Automatic Machines and Robots*, Berlin Heidelberg: Springer, 1. Edn., ISBN 3540856285
- Bickendorf (2014) Bickendorf, Jobst, 2014. Robotic Welding of Ship-Subassemblies with fully automatic Offline-Programming, In: *Proceedings of 41st International Symposium on Robotics ISR/Robotik*, 2-3 June 2014, Munich Germany, pp. 428–434, ISBN 978-3-8007-3601-0

-
- Biggs and Macdonald (2003) Biggs, Geoffrey and Macdonald, Bruce, 2003. A Survey of Robot Programming Systems, In: *Proceedings of the Australasian Conference on Robotics and Automation (CSIRO)*, 1-3 December 2003, Brisbane Queensland
- Billard, Calinon et al. (2008) Billard, Aude, Calinon, Sylvain, Dillmann, Rüdiger, and Schaal, Stefan, 2008. Robot Programming by Demonstration, In: Siciliano, Bruno and Khatib, Oussama (Eds.), *Springer Handbook of Robotics*, Berlin Heidelberg: Springer, pp. 1371–1394, ISBN 978-3-540-30301-5
- Björkenstam, Spensieri et al. (2014) Björkenstam, Staffan, Spensieri, Domenico, Carlson, Johan S., Bohlin, Robert, and Gleeson, Daniel, 2014. Efficient Sequencing of Industrial Robots through Optimal Control, *Procedia CIRP* **Vol. 23**(C) pp. 194–199, doi:10.1016/j.procir.2014.10.094
- Bohlin and Kavraki (2000) Bohlin, Robert and Kavraki, Lydia, 2000. Path planning using lazy PRM, In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings*, 24-27 April 2000, San Francisco CA USA, pp. 521–528, ISBN 0-7803-5886-4, doi:10.1109/ROBOT.2000.844107
- Bolmsjo (2014) Bolmsjo, Gunnar, 2014. Reconfigurable and Flexible Industrial Robot Systems, *Advances in Robotics & Automation* **Vol. 03**(01), doi:10.4172/2168-9695.1000117
- Bolmsjö, Olsson et al. (2002) Bolmsjö, Gunnar, Olsson, Magnus, and Cederberg, Per, 2002. Robotic arc welding – trends and developments for higher autonomy, *Industrial Robot* **Vol. 29**(2) pp. 98–104, doi:10.1108/01439910210419088
- Bone and Elbestawi (1992) Bone, Gary M. and Elbestawi, M. A., 1992. Sensing and control for automated robotic edge deburring, In: *Proceedings of the International Conference on Industrial Electronics, Control, Instrumentation, and Automation*, 13 November 1992, San Diego CA USA, pp. 927–933, doi:10.1109/IECON.1992.254481
- Borghesan, Scioni et al. (2016) Borghesan, Gianni, Scioni, Enea, Kheddar, Abderrahmane, and Bruyninckx, Herman, 2016. Introducing Geometric Constraint Expressions Into Robot Constrained Motion Specification and Control, *IEEE Robotics and Automation Letters* **Vol. 1**(2) pp. 1140–1147, doi:10.1109/LRA.2015.2506119

-
- Budak and Tekeli (2005) Budak, Erhan and Tekeli, A., 2005. Maximizing Chatter Free Material Removal Rate in Milling through Optimal Selection of Axial and Radial Depth of Cut Pairs, *CIRP Annals - Manufacturing Technology* **Vol. 54**(1) pp. 353–356, doi:10.1016/S0007-8506(07)60121-8
- Carlson, Spensieri et al. (2014) Carlson, Johan S., Spensieri, Domenico, Wärmefjord, Kristina, Segeborn, Johan, and Söderberg, Rikard, 2014. Minimizing Dimensional Variation and Robot Traveling Time in Welding Stations, *Procedia CIRP* **Vol. 23**(C) pp. 77–82, doi:10.1016/j.procir.2014.03.199
- Caro, Dumas et al. (2013) Caro, Stéphane, Dumas, Claire, Garnier, Sebastien, and Furet, Benoit, 2013. Workpiece Placement Optimization in Robotic-based Manufacturing, *IFAC Proceedings Volumes* **Vol. 46**(9) pp. 819–824, doi:10.3182/20130619-3-RU-3018.00211
- Cheah, Hou et al. (2010) Cheah, Chien Chern, Hou, Saing Paul, Zhao, Yu, and Slotine, Jean J. E., 2010. Adaptive Vision and Force Tracking Control for Robots With Constraint Uncertainty, *IEEE/ASME Transactions on Mechatronics* **Vol. 15**(3) pp. 389–399, doi:10.1109/TMECH.2009.2027115
- Chen, Fuhlbrigge et al. (2008) Chen, Heping, Fuhlbrigge, Thomas, and Li, Xiongzi, 2008. Automated industrial robot path planning for spray painting process: A review, In: *IEEE International Conference on Automation Science and Engineering (CASE)*, 23-26 August 2008, Arlington VA USA, pp. 522–527, doi:10.1109/COASE.2008.4626515
- Chen and Dong (2013) Chen, Yonghua and Dong, Fenghua, 2013. Robot machining: recent development and future research issues, *The International Journal of Advanced Manufacturing Technology* **Vol. 66**(9) pp. 1489–1497, doi:10.1007/s00170-012-4433-4
- Cho, Kim et al. (2017) Cho, Youngsu, Kim, Munyu, Cheong, Joono, Do, Hyunmin, and Kyung, Jinho, 2017. Simultaneous identification of kinematic screw and joint compliance of elastic robot manipulators using deflected circular trajectories, In: *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 3-7 July 2017, Munich Germany, pp. 382–387, doi:10.1109/AIM.2017.8014047

-
- Choi and Amir (2009) Choi, Jaesik and Amir, Eyal, 2009. Combining planning and motion planning, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 12-17 May 2009, Kobe Japan, pp. 238–244, doi:10.1109/ROBOT.2009.5152872
- Choi, Jung et al. (2015) Choi, SangSu, Jung, Kiwook, and Noh, Sang Do, 2015. Virtual reality applications in manufacturing industries: Past research, present findings, and future directions, *Concurrent Engineering* **Vol. 23**(1) pp. 40–63, doi:10.1177/1063293X14568814
- Choset, Lynch et al. (2005) Choset, Howie, Lynch, Kevin M., Hutchinson, Seth, Kantor, George A., Burgard, Wolfram, Kavraki, Lydia E., and Thrun, Sebastian, 2005. *Principles of Robot Motion: Theory, Algorithms, and Implementations*, Pittsburgh PA: MIT Press, ISBN 0262033275
- Chou, You et al. (2007) Chou, Wusheng, You, Liang, and Wang, Tianmiao, 2007. Automatic Path Planning for Welding Robot Based on Reconstructed Surface Model, In: *Robotic Welding, Intelligence and Automation*, Berlin Heidelberg: Springer, pp. 153–161, ISBN 978-3-540-73374-4, doi:10.1007/978-3-540-73374-4_17
- Chung and Kim (2006) Chung, Jae H. and Kim, Changhoon, 2006. Modeling and Control of a New Robotic Deburring System, *Robotica* **Vol. 24**(2) pp. 229–237, doi:10.1017/S0263574705002067
- Coleman, Sucan et al. (2014) Coleman, David, Sucan, Ioan Alexandru, Chitta, Sachin, and Correll, Nikolaus, 2014. Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study, *Computing Research Repository (CoRR)* URL <https://arxiv.org/pdf/1404.3785.pdf>, retrieved: 22.03.18
- COMET (2013) COMET, 2013. COMET Technologies: Programming and Simulation Environment for Industrial Robots, URL <http://www.cometproject.eu/publications/delcam-psir.pdf>, retrieved: 03.09.17
- Corke (2011) Corke, Peter, 2011. *Robotics, Vision and Control - Fundamental Algorithms in MATLAB®*, Springer Tracts in Advanced Robotics, Vol. 73, Berlin Heidelberg: Springer, ISBN 978-3-642-20143-1

-
- Craig (1989) Craig, John J., 1989. *Introduction to Robotics: Mechanics and Control*, Boston MA USA: Addison-Wesley Longman Publishing, 2. Edn., ISBN 0201095289
- Csiszar, Drust et al. (2012) Csiszar, Akos, Drust, Manuel, Dietz, Thomas, Verl, Alexander, and Brisan, Cornel, 2012. Dynamic and Interactive Path Planning and Collision Avoidance for an Industrial Robot Using Artificial Potential Field Based Method, In: Jabłoński, Ryszard and Březina, Tomáš (Eds.), *Mechatronics*, Berlin Heidelberg: Springer, pp. 413–421, ISBN 978-3-642-23243-5, doi:10.1007/978-3-642-23244-2_50
- Cutting-Decelle, Young et al. (2007) Cutting-Decelle, Anne-Francoise, Young, Robert I. M., Michel, John J., Grangel, Reyes, Cardinal, J. Le, and Bourey, J. P., 2007. ISO 15531 mandate: A Product-process-resource based Approach for Managing Modularity in Production Management, *Concurrent Engineering* **Vol. 15**(2) pp. 217–235, doi:10.1177/1063293X07079329
- Dawande, Geismar et al. (2007) Dawande, Milind W., Geismar, H. Neil, Sethi, Suresh P., and Sriskandarajah, Chelliah, 2007. *Throughput optimization in robotic cells*, International Series in Operations Research & Management Science, Vol. 101, New York USA: Springer Science & Business Media, ISBN 0-387-70987-8
- De Schutter, De Laet et al. (2007) De Schutter, Joris, De Laet, Tinne, Rutgeerts, Johan, Decré, Wilm, Smits, Ruben, Aertbeliën, Erwin, Claes, Kasper, and Bruyninckx, Herman, 2007. Constraint-Based Task Specification and Estimation for Sensor-Based Robot Systems in the Presence of Geometric Uncertainty, *The International Journal of Robotics Research* **Vol. 26**(5) pp. 433–455, doi:10.1177/027836490707809107
- Delmia (2017) Delmia, 2017. DPM Assembly: the Complete Digital Assembly Process Planning solution, URL <http://www.3ds.com/fileadmin/PRODUCTS/DELMIA/PDF/Brochures/delmia-dpm-assembly.pdf>, retrieved: 21.03.18
- Denkena, Wörn et al. (2004) Denkena, Berend, Wörn, H., Hein, B., Apitz, R., Kowalski, P., and Mages, D., 2004. Vereinfachte Programmierung von Industrierobotern: Industrieroboter, CAM (Computer

-
- Aided Manufacturing), Virtual Reality, *wt Werkstattstechnik online* **Vol. 94**(9) pp. 442–446
- Devaurs, Siméon et al. (2013) Devaurs, Didier, Siméon, Thierry, and Cortés, Juan, 2013. Enhancing the transition-based RRT to deal with complex cost spaces, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 6-10 May 2013, Karlsruhe Germany, pp. 4120–4125, doi:10.1109/ICRA.2013.6631158
- Devillers and Guigue (2002) Devillers, Olivier and Guigue, Philippe, 2002. Faster Triangle-Triangle Intersection Tests: INRIA, URL <https://hal.inria.fr/inria-00072100>, retrieved: 21.03.18
- Diankov (2010) Diankov, Rosen, 2010. *Automated Construction of Robotic Manipulation Programs*, Pittsburgh PA, Carnegie Mellon Univ., Diss., 2010, URL http://www.programmingvision.com/rosen_diankov_thesis.pdf, retrieved: 21.03.18
- Diankov, Ueda et al. (2011) Diankov, Rosen, Ueda, Ryohei, Okada, Kei, and Saito, Hajime, 2011. Collada: An open standard for robot file formats, In: *Proceedings of the 29th Annual Conference of the Robotics Society of Japan*, 7-9 September 2011, Shibaur Japan
- Diaz Posada, Dietz et al. (2016) Diaz Posada, Julian Ricardo, Dietz, Thomas, Ockert, Philip, Kuss, Alexander, Hägele, Martin, and Verl, Alexander, 2016. Automatic Optimal Motion Generation for Robotic Manufacturing Processes: Optimal Collision Avoidance in Robotic Welding, In: *IEEE International Conference on Automation Science and Engineering (CASE)*, 21-24 August 2016, Fort Worth Texas USA, pp. 154–161, ISBN 9781509024100, doi:10.1109/COASE.2016.7743374
- Diaz Posada, Kumar et al. (2016) Diaz Posada, Julian Ricardo, Kumar, Shivaram, Kuss, Alexander, Schneider, Ulrich, Drust, Manuel, Dietz, Thomas, and Verl, Alexander, 2016. Automatic Programming and Control for Robotic Deburring, In: *Proceedings of ISR 2016: 47th International Symposium on Robotics*, 21-22 June 2016, Munich Germany, pp. 1–8, ISBN 978-3-8007-4231-8
- Diaz Posada, Schneider et al. (2016) Diaz Posada, Julian Ricardo, Schneider, Ulrich, Pidan, Sergej, Geravand, Milad, Stelzer, Patrick, and Verl, Alexander, 2016. High Accurate Robotic Drilling with External Sensor and Compliance Model-Based

-
- Compensation, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 16-21 May 2016, Stockholm Sweden, pp. 3901–3907, doi:10.1109/ICRA.2016.7487579
- Diaz Posada, Schneider et al. (2017) Diaz Posada, Julian Ricardo, Schneider, Ulrich, Sridhar, Arjun, and Verl, Alexander, 2017. Automatic Motion Generation for Robotic Milling Optimizing Stiffness with Sample-Based Planning, *MDPI Machines* **Vol. 5**(1), doi:10.3390/machines5010003
- Diaz Posada, Mukherjee et al. (2018) Diaz Posada, Julian Ricardo, Mukherjee, Poulastya, and Verl, Alexander, 2018. Automatic Close-optimal Workpiece Positioning for Robotic Manufacturing., In: *The 51st CIRP Conference on Manufacturing Systems (CIRP CMS 2018)*, 16-18 May, Stockholm Sweden, pp. 277–284, doi:10.1016/j.procir.2018.03.142
- Dietz, Schneider et al. (2012) Dietz, Thomas, Schneider, Ulrich, Barho, Marc, Oberer-Treitz, Susanne, Drust, Manuel, Hollmann, Rebecca, and Hägele, Martin, 2012. Programming System for Efficient Use of Industrial Robots for Deburring in SME Environments, In: *ROBOTIK 2012; 7th German Conference on Robotics*, 21-22 May 2012, Munich Germany, pp. 1–6, ISBN 978-3-8007-3418-4
- Dietz, Pott et al. (2015) Dietz, Thomas, Pott, Andreas, Hägele, Martin, and Verl, Alexander, 2015. Knowledge-based cost engineering for industrial robot systems, In: *IEEE International Conference on Automation Science and Engineering (CASE)*, 24-28 August 2015, Gothenburg Sweden, pp. 1200–1205, doi:10.1109/CoASE.2015.7294261
- Ding, Pan et al. (2015a) Ding, Dong-Hong, Pan, Zeng-Xi, Dominic, Cuiuri, and Li, Hui-Jun, 2015. Process Planning Strategy for Wire and Arc Additive Manufacturing, In: Tarn, Tzyh-Jong, Chen, Shan-Ben, and Chen, Xiao-Qi (Eds.), *Robotic Welding, Intelligence and Automation: RWIA '2014*, Cham: Springer International Publishing, pp. 437–450, ISBN 978-3-319-18997-0, doi:10.1007/978-3-319-18997-0_37
- Ding, Pan et al. (2015b) Ding, Donghong, Pan, Zengxi, Cuiuri, Dominic, and Li, Huijun, 2015. Process planning for robotic wire and arc additive manufacturing, In: *IEEE 10th Conference on Industrial Electronics and Applications (ICIEA)*, 15-17 June

-
- 2015, Auckland New Zealand, pp. 2000–2003,
doi:10.1109/ICIEA.2015.7334441
- Djuric and ElMaraghy (2009) Djuric, A. M. and ElMaraghy, W. H., 2009. Unified Dynamic and Control Models for Reconfigurable Robots, In: ElMaraghy, Hoda A. (Ed.), *Changeable and Reconfigurable Manufacturing Systems*, London: Springer, pp. 147–161, ISBN 978-1-84882-067-8, doi:10.1007/978-1-84882-067-8_8
- Doan, Tao et al. (2016) Doan, Ngoc Chi Nam, Tao, Pey Yuen, and Lin, Wei, 2016. Optimal redundancy resolution for robotic arc welding using modified particle swarm optimization, In: *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 12-15 July 2016, Banff Canada, pp. 554–559, doi:10.1109/AIM.2016.7576826
- Dobson and Bekris (2013) Dobson, Andrew and Bekris, Kostas E., 2013. Improving sparse roadmap spanners, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 6-10 May 2013, Karlsruhe Germany, pp. 4106–4111, doi:10.1109/ICRA.2013.6631156
- Dobson, Krontiris et al. (2013) Dobson, Andrew, Krontiris, Athanasios, and Bekris, Kostas E., 2013. Sparse Roadmap Spanners, In: Frazzoli, Emilio, Lozano-Perez, Tomas, Roy, Nicholas, and Rus, Daniela (Eds.), *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*, Berlin Heidelberg: Springer, pp. 279–296, ISBN 978-3-642-36279-8, doi:10.1007/978-3-642-36279-8_17
- Dolgui and Pashkevich (2004) Dolgui, Alexandre and Pashkevich, Anatol, 2004. Cluster-level operations planning in arc-welding robot cell with positioning table, URL <http://hal-emse.ccsd.cnrs.fr/emse-00709877>, retrieved: 22.03.18
- Dolgui, Pashkevich et al. (2005) Dolgui, Alexandre, Pashkevich, Anatoly, and Semkin, Konstantin, 2005. Manufacturing process planning for robotic arc-welding station with positioning table, In: *The 6th IEEE International Symposium on Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing*, 19-21 July 2005, Montreal Canada, pp. 1–6, doi:10.1109/ISATP.2005.1511441

-
- Drath, Lüder et al. (2008) Drath, Rainer, Lüder, Arndt, Peschke, Jörn, and Hundt, Lorenz, 2008. AutomationML - the glue for seamless automation engineering, In: *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 15-18 September 2008, Hamburg Germany, pp. 616–623, doi:10.1109/ETFA.2008.4638461
- Dumas, Caro et al. (2010) Dumas, Claire, Caro, Stéphane, Cherif, Mehdi, Garnier, Sebastien, and Furet, Benoit, 2010. A methodology for joint stiffness identification of serial robots, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 18-22 October 2010, Taipei Taiwan, pp. 464–469, doi:10.1109/IROS.2010.5652140
- Dumas, Caro et al. (2011) Dumas, Claire, Caro, Stéphane, Garnier, Sébastien, and Furet, Benoît, 2011. Joint stiffness identification of six-revolute industrial serial robots, *Robotics and Computer-Integrated Manufacturing* **Vol. 27**(4) pp. 881–888, doi:10.1016/j.rcim.2011.02.003
- Eggers, Hansen et al. (2014) Eggers, Kai, Hansen, Christian, Kotlarski, Jens, and Ortmaier, Tobias, 2014. Reduction of Energy Consumption by Optimized Robot Cell Design, In: *Symposium on Automated Systems and Technologies*, 15-16 October 2014, Garbsen: Tewiss, pp. 65–70, ISBN 978-3-944586-84-7
- Elbanhawî and Simic (2014) Elbanhawî, Mohamed and Simic, Milan, 2014. Sampling-Based Robot Motion Planning: A Review, *IEEE Access* **Vol. 2** pp. 56–77, doi:10.1109/ACCESS.2014.2302442
- Erdős, Kemény et al. (2013) Erdős, Gábor, Kemény, Zsolt, Kovács, András, and Váncza, József, 2013. Planning of Remote Laser Welding Processes, *Procedia CIRP* **Vol. 7**(C) pp. 222–227, doi:10.1016/j.procir.2013.05.038
- Erdős, Kardos et al. (2016) Erdős, Gábor, Kardos, Csaba, Kemény, Zsolt, Kovács, András, and Váncza, József, 2016. Process planning and offline programming for robotic remote laser welding systems, *International Journal of Computer Integrated Manufacturing* **Vol. 29**(12) pp. 1287–1306, doi:10.1080/0951192X.2015.1033753
- Erez, Tassa et al. (2015) Erez, Tom, Tassa, Yuval, and Todorov, Emanuel, 2015. Simulation tools for model-based robotics: Comparison of Bullet, Havok, MuJoCo, ODE and PhysX, In: *IEEE*

International Conference on Robotics and Automation (ICRA), 26-30 May 2015, Seattle WA USA, pp. 4397–4404, ISBN 978-1-4799-6923-4, doi:10.1109/ICRA.2015.7139807

- Ericsson and Nylén (2007) Ericsson, Mikael and Nylén, P., 2007. A Look at the Optimization of Robot Welding Speed Based on Process Modeling, *Welding Journal* **Vol. 86** pp. 238–244
- Feldmann, Schmuck et al. (2008) Feldmann, Klaus, Schmuck, T., Brossog, M., and Dreyer, J., 2008. Beschreibungsmodell zur Planung von Produktionssystemen, *wt Werkstattstechnik online* **Vol. 98**(3) pp. 156–162
- Ferguson (2013) Ferguson, Michael, 2013. Search-based planning library (SBPL): Package Summary, URL <http://wiki.ros.org/sbpl>, retrieved: 22.03.18
- Gammell, Srinivasa et al. (2014) Gammell, Jonathan D., Srinivasa, Siddhartha S., and Barfoot, Timothy D., 2014. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 14-18 September 2014, Chicago IL USA, pp. 2997–3004, doi:10.1109/IROS.2014.6942976
- Gammell, Srinivasa et al. (2015) Gammell, Jonathan D., Srinivasa, Siddhartha S., and Barfoot, Timothy D., 2015. Batch Informed Trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 26-30 May 2015, Seattle WA USA, pp. 3067–3074, ISBN 978-1-4799-6923-4, doi:10.1109/ICRA.2015.7139620
- Gerdts, Henrion et al. (2012) Gerdts, Matthias, Henrion, René, Hömberg, Dietmar, and Landry, Chantal, 2012. Path planning and collision avoidance for robots, *Numerical Algebra, Control and Optimization* **Vol. 2**(3) pp. 437–463, doi:10.3934/naco.2012.2.437
- Ghallab, Nau et al. (2004) Ghallab, Malik, Nau, Dana, and Traverso, Paolo, 2004. *Automated Planning: Theory & Practice*, San Francisco CA USA: Morgan Kaufmann Publishers Inc., ISBN 1558608567
- Gipson, Gupta et al. (2001) Gipson, Ian, Gupta, Kamal, and Greenspan, Michael, 2001. MPK: An open extensible motion planning kernel, *Journal*

-
- of Robotic Systems* **Vol. 18**(8) pp. 433–443,
doi:10.1002/rob.1035
- Gomeringer (2014) Gomeringer, Roland, 2014. *Tabellenbuch Metall*,
Haan-Gruiten: Europa-Lehrmittel, 46. Edn., ISBN
3808512709
- Guigue and Devillers (2003) Guigue, Philippe and Devillers, Olivier, 2003. Fast and
Robust Triangle-Triangle Overlap Test Using Orientation
Predicates, *Journal of graphics tools* **Vol. 8**(1) pp. 39–52,
doi:10.1080/10867651.2003.10487580, URL
<https://hal.inria.fr/hal-00795042>
- Haage, Halbauer et al. (2014) Haage, Mathias, Halbauer, Marcel, Lehmann, Christian,
and Staedter, J. Philipp, 2014. Increasing Robotic
Machining Accuracy Using Offline Compensation Based on
Joint-Motion Simulation, In: *Proceedings of 41st
International Symposium on Robotics ISR/Robotik*, 2-3 June
2014, Munich Germany, pp. 1–8, ISBN 978-3-8007-3601-0
- Hägele, Nilsson et al. (2008) Hägele, Martin, Nilsson, Klas, and Pires, J. Norberto, 2008.
Industrial Robotics, In: Siciliano, Bruno and Khatib,
Oussama (Eds.), *Springer Handbook of Robotics*, Berlin
Heidelberg: Springer, pp. 963–986, ISBN 978-3-540-30301-5
- Halperin, Latombe et al. (2000) Halperin, D., Latombe, J.-C., and Wilson, R. H., 2000. A
General Framework for Assembly Planning: The Motion
Space Approach, *Algorithmica* **Vol. 26**(3-4) pp. 577–601,
doi:10.1007/s004539910025
- Hartfuss (1996) Hartfuss, Christoph, 1996. *Wissensbasierte Programmierung
von Industrierobotern zum Schutzgasschweißen im
Stahlhochbau*, Berlin Heidelberg: Springer, ISBN
3-540-60661-0, Stuttgart, Univ., Diss., 1996
- Hein (2003) Hein, Björn, 2003. *Automatische Offline-Programmierung
von Industrierobotern in der virtuellen Welt*, Waabs:
GCA-Verlag, ISBN 9783898631419, Karlsruher Institut für
Technologie, Univ., Diss., 2003
- Hein, Salonia et al. (2001) Hein, Björn, Salonia, Marcos, and Wörn, Heinz, 2001.
Automated generated collision-free time optimized robot
movements in industrial environments based on rounding,
In: *Proceedings of the 2001 IEEE International Symposium
on Assembly and Task Planning (ISATP2001)*. *Assembly
and Disassembly in the Twenty-first Century*, 29 May 2001,

-
- Fukuoka Japan, pp. 110–115, ISBN 0-7803-7004-X, doi:10.1109/ISATP.2001.928975
- Helms (2007) Helms, Evert, 2007. *Roboterbasierte Bahnführungsunterstützung von industriellen Handhabungs- und Bearbeitungsprozessen*, Heimsheim: Jost-Jetter-Vlg, ISBN 978-3-939890-04-1, Stuttgart, Univ., Diss., 2007
- Hollmann, Rost et al. (2010) Hollmann, Rebecca, Rost, Arne, Hägele, Martin, and Verl, Alexander, 2010. A HMM-based approach to learning probability models of programming strategies for industrial robots, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 3-8 May 2010, Anchorage Alaska USA, pp. 2965–2970, doi:10.1109/ROBOT.2010.5509888
- Hua, Zander et al. (2016) Hua, Yingbing, Zander, Stefan, Bordinon, Mirko, and Hein, Björn, 2016. From AutomationML to ROS: A model-driven approach for software engineering of industrial robotics using ontological reasoning, In: *IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 6-9 September, Berlin Germany, pp. 1–8, doi:10.1109/ETFA.2016.7733579
- Huo and Baron (2005) Huo, Liguu and Baron, Luc, 2005. Kinematic Inversion of Functionally-Redundant Serial Manipulators: Application to Arc-Welding, *Transactions of the Canadian Society for Mechanical Engineering* **Vol. 29**(4) pp. 679–690
- Huo and Baron (2008) Huo, Liguu and Baron, Luc, 2008. The joint-limits and singularity avoidance in robotic welding, *Industrial Robot: The International Journal of Robotics Research And Application* **Vol. 35**(5) pp. 456–464
- IFR Statistical Department (2014) IFR Statistical Department (Ed.), 2014. *World Robotics 2014 Industrial Robots*, ISBN 978-3-8163-0665-8
- IFR Statistical Department (2015) IFR Statistical Department (Ed.), 2015. *World Robotics 2015 Industrial Robots*, ISBN 978-3-8163-0679-5
- IFR Statistical Department (2016) IFR Statistical Department (Ed.), 2016. *World Robotics 2016 Industrial Robots*, ISBN 978-3-8163-0693-1
- IFR Statistical Department (2017) IFR Statistical Department (Ed.), 2017. *World Robotics 2017 Industrial Robots*, ISBN 978-3-8163-0713-6
- Iglesias, Sebastián et al. (2015) Iglesias, I., Sebastián, M. A., and Ares, J. E., 2015. Overview of the State of Robotic Machining: Current

Situation and Future Potential, *Procedia Engineering* **Vol. 132** pp. 911–917, doi:10.1016/j.proeng.2015.12.577

International Organization for Standardization (1992)

International Organization for Standardization, 1992.
Welded, brazed and soldered joints - Symbolic
representation on drawings

International Organization for Standardization (2012)

International Organization for Standardization, 2012. ISO
8373:2012(E/F), Robots and Robotic Devices - Vocabulary.
Geneva: ISO copyright office

Ivaldi, Padois et al. (2014)

Ivaldi, Serena, Padois, Vincent, and Nori, Francesco, 2014.
Tools for dynamics simulation of robots: a survey based on
user feedback, *Computing Research Repository (CoRR)* URL
<https://arxiv.org/abs/1402.7050>, retrieved: 22.03.18

Jaillet, Cortés et al. (2008)

Jaillet, Léonard, Cortés, Juan, and Siméon, Thierry, 2008.
Transition-based RRT for path planning in continuous cost
spaces, In: *IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)*, 22-26 September
2008, Nice France, pp. 2145–2150,
doi:10.1109/IROS.2008.4650993

Jaillet, Cortés et al. (2016)

Jaillet, Leonard, Cortés, Juan, and Simeon, Thierry, 2016.
Sampling-based path planning on configuration-space
costmaps, *IEEE journal on robotics and automation*
Vol. 26(4) pp. 635–646, doi:10.1109/TRO.2010.2049527

Janson, Schmerling et al. (2015)

Janson, Lucas, Schmerling, Edward, Clark, Ashley, and
Pavone, Marco, 2015. Fast marching tree: A fast marching
sampling-based method for optimal motion planning in
many dimensions, *The International Journal of Robotics
Research* **Vol. 34**(7) pp. 883–921,
doi:10.1177/0278364915577958

Jayaweera and Webb (2010)

Jayaweera, Nirosh and Webb, Phil, 2010. Measurement
Assisted Robotic Edge Deburring of Aero Engine
Components, In: *Proceedings of the 9th WSEAS
international conference on Signal processing, robotics and
automation*, 20-22 February, Nottingham UK, pp. 133–138,
ISBN 978-960-474-157-1

Jayaweera and Webb (2011)

Jayaweera, Nirosh and Webb, Phil, 2011. Robotic edge
profiling of complex components, *Industrial Robot: An*

-
- International Journal* **Vol. 38**(1) pp. 38–47,
doi:10.1108/01439911111097832
- Jiménez, Thomas et al. (2001) Jiménez, Pablo, Thomas, Federico, and Torras, Carme, 2001. 3D collision detection: a survey, *Computers & Graphics* **Vol. 25**(2) pp. 269–285, doi:10.1016/S0097-8493(00)00130-8
- Jonsson (2013) Jonsson, Marie, 2013. *On Manufacturing Technology as an Enabler of Flexibility: Affordable Reconfigurable Tooling and Force-Controlled Robotics*, Linköping Sweden: LiU-Tryck, ISBN 9789175196916, Linköping Univ., Diss., 2013
- Juang (1998) Juang, Jih-Gau, 1998. Collision avoidance using potential fields, *International Journal of Robotics Research and Application* **Vol. 25**(6) pp. 408–415,
doi:10.1108/01439919810240216
- Kah, Shrestha et al. (2015) Kah, P., Shrestha, M., Hiltunen, E., and Martikainen, J., 2015. Robotic arc welding sensors and programming in industrial applications, *International Journal of Mechanical and Materials Engineering* **Vol. 10**(1) p. 13,
doi:10.1186/s40712-015-0042-y
- Karaman and Frazzoli (2011) Karaman, Sertac and Frazzoli, Emilio, 2011. Sampling-based Algorithms for Optimal Motion Planning, *International Journal of Robotics Research* **Vol. 30**(7) pp. 846–894, doi:10.1177/0278364911406761
- Karim and Verl (2013) Karim, Ali and Verl, Alexander, 2013. Challenges and obstacles in robot-machining, In: *44th International Symposium on Robotics (ISR)*, 24-26 Oct 2013, Seoul South Korea, pp. 1–4, doi:10.1109/ISR.2013.6695731
- Karim, Hitzer et al. (2017) Karim, Ali, Hitzer, Julian, Lechler, Armin, and Verl, Alexander, 2017. Analysis of the dynamic behavior of a six-axis industrial robot within the entire workspace in respect of machining tasks, In: *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 3-7 July 2017, Munich Germany, pp. 670–675,
doi:10.1109/AIM.2017.8014094
- Kavraki (2017) Kavraki, Lab, 2017. Open Motion Planning Library: A Primer: Rice Univ., URL
http://ompl.kavrakilab.org/OMPL_Primer.pdf,
retrieved: 22.03.18

-
- Kavraki, Svestka et al. (1996) Kavraki, Lydia, Svestka, Petr, Latombe, Jean-Claude, and Overmars, Mark, 1996. Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces, In: *IEEE Transactions on Robotics and Automation*, Minneapolis Minnesota, August 1996, pp. 566–580, doi:10.1109/70.508439
- Kim, Rhee et al. (1999) Kim, Pyunghyun, Rhee, Sehun, and Lee, Chang Hee, 1999. Automatic teaching of welding robot for free-formed seam using laser vision sensor, *Optics and Lasers in Engineering* **Vol. 31**(3) pp. 173–182, doi:10.1016/S0143-8166(99)00024-X
- Klimchik, Wu et al. (2013) Klimchik, Alexandr, Wu, Yier, Dumas, Claire, Caro, Stéphane, Furet, Benoît, and Pashkevich, Anatol, 2013. Identification of geometrical and elastostatic parameters of heavy industrial robots, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 6-10 May 2013, Karlsruhe Germany, doi:10.1109/ICRA.2013.6631098
- Klimchik, Bondarenko et al. (2014) Klimchik, Alexandr, Bondarenko, Dmitry, Pashkevich, Anatol, Briot, Sébastien, and Furet, Benoît, 2014. Compliance Error Compensation in Robotic-Based Milling, In: Ferrier, Jean-Louis, Alain, Bernard, Oleg, Gusikhin, and Kurosh, Madani (Eds.), *Informatics in Control, Automation and Robotics: 9th International Conference, ICINCO 2012 Rome, Italy, July 28-31, 2012*, Cham: Springer International Publishing, pp. 197–216, ISBN 978-3-319-03500-0, doi:10.1007/978-3-319-03500-0_13
- Klingstam and Gullander (1999) Klingstam, Pär and Gullander, Per, 1999. Overview of simulation tools for computer-aided production engineering, *Computers in Industry* **Vol. 38**(2) pp. 173–186, doi:10.1016/S0166-3615(98)00117-1
- Kockara, Halic et al. (2007) Kockara, Sinan, Halic, Tansel, Iqbal, Kamran, Bayrak, Coskun, and Rowe, Richard, 2007. Collision detection: A survey, In: *IEEE International Conference on Systems, Man and Cybernetics*, 7-10 October 2007, Montreal Canada, pp. 4046–4051, doi:10.1109/ICSMC.2007.4414258
- Kröger, Finkemeyer et al. (2004) Kröger, Torsten, Finkemeyer, Bernd, and Wahl, Friedrich M., 2004. A task frame formalism for practical implementations, In: *IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans LA USA,

-
- 26.4.-1.5. 2004, pp. 5218–5223,
doi:10.1109/ROBOT.2004.1302546
- Krug (2013) Krug, Stefan, 2013. *Automatische Konfiguration von Robotersystemen (Plug&Produce)*, Munich: Herbert Utz Verlag, ISBN 9783831642434, Munich Univ., Diss., 2013
- Kuffner and Lavalley (2000) Kuffner, James Jr. and Lavalley, Steven M., 2000. RRT-Connect: An efficient approach to single-query path planning, In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings*, 24-27 April 2000, San Francisco CA USA, pp. 995–1001, ISBN 0-7803-5886-4, doi:10.1109/ROBOT.2000.844730
- Kuhlenkoetter, Schyja et al. (2010) Kuhlenkoetter, Bernd, Schyja, Adrian, Hypki, Alfred, and Miegel, Volker, 2010. Robot Workcell Simulation with AutomationML Support - An Element of the CAx-Tool Chain in Industrial Automation, In: *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, 7-9 June 2010, Munich Germany, pp. 1–7, ISBN 978-3-8007-3273-9
- Kuss, Diaz Posada et al. (2016) Kuss, Alexander, Diaz Posada, Julian Ricardo, Hollmann, Rebecca, Dietz, Thomas, and Hägele, Martin, 2016. Manufacturing knowledge for industrial robot systems: Review and synthesis of model architecture, In: *IEEE International Conference on Automation Science and Engineering (CASE)*, 21-24 August 2016, Fort Worth Texas USA, pp. 348–354, ISBN 9781509024100, doi:10.1109/COASE.2016.7743427
- Kuss, Drust et al. (2016) Kuss, Alexander, Drust, Manuel, and Verl, Alexander, 2016. Detection of workpiece shape deviations for tool path adaptation in robotic deburring systems, In: *49th CIRP Conference on Manufacturing Systems (CIRP-CMS 2016)*, 25-27 May 2016, Stuttgart, doi:10.1016/j.procir.2016.11.094
- Kuss, Dietz et al. (2017) Kuss, Alexander, Dietz, Thomas, Ksensow, Konstantin, and Verl, Alexander, 2017. Manufacturing Task Description for Robotic Welding and Automatic Feature Recognition on Product CAD Models, *Procedia CIRP* **Vol. 60**(C) pp. 122–127, doi:10.1016/j.procir.2017.01.045
- Kuss, Ksensow et al. (2017) Kuss, Alexander, Ksensow, Konstantin, and Verl, Alexander, 2017. Deformable shape registration using surface-based

-
- free-form deformations in robotic welding, In: *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 3-7 July 2017, Munich Germany, pp. 885–890, doi:10.1109/AIM.2017.8014130
- Laet, Decre et al. (2007) Laet, Tinne De, Decre, Wilm, Rutgeerts, Johan, Bruyninckx, Herman, and Schutter, Joris De, 2007. An application of constraint-based task specification and estimation for sensor-based robot systems, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 29.10.-2.11.2007, San Diego CA USA, pp. 1658–1664, ISBN 978-1-4244-0911-2, doi:10.1109/IROS.2007.4399016
- Laet, Bellens et al. (2013a) Laet, Tinne De, Bellens, Steven, Bruyninckx, Herman, and Schutter, Joris De, 2013. Geometric Relations Between Rigid Bodies (Part 1): From Semantics to Software, *IEEE Robotics and Automation Magazine* **Vol. 20**(2) pp. 91–102, doi:10.1109/MRA.2012.2230501
- Laet, Bellens et al. (2013b) Laet, Tinne De, Bellens, Steven, Smits, Ruben, Aertbelien, Erwin, Bruyninckx, Herman, and Schutter, Joris De, 2013. Geometric Relations Between Rigid Bodies (Part 2): Semantics for Standardization, *IEEE Robotics and Automation Magazine* **Vol. 20**(1) pp. 84–93, doi:10.1109/MRA.2012.2205652
- Landry, Gerdtts et al. (2014) Landry, Chantal, Gerdtts, Matthias, Henrion, René, Hömberg, Dietmar, and Welz, Wolfgang, 2014. Collision-Free Path Planning of Welding Robots, In: Magnus, Fontes, Günther, Michael, and Marheineke, Nicole (Eds.), *Progress in Industrial Mathematics at ECMI 2012*, Cham: Springer International Publishing, pp. 251–256, ISBN 978-3-319-05365-3, doi:10.1007/978-3-319-05365-3_34
- Larkin, Short et al. (2016) Larkin, Nathan, Short, Andrew, Pan, Zengxi, and Duin, Stephen van, 2016. Automatic program generation for welding robots from CAD, In: *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 12-15 July 2016, Banff Canada, pp. 560–565, doi:10.1109/AIM.2016.7576827
- Larsen, Pham et al. (2015) Larsen, Lars, Pham, Van-Lang, Kim, Jonghwa, and Kupke, Michael, 2015. Collision-free path planning of industrial cooperating robots for aircraft fuselage production, In:

-
- IEEE International Conference on Robotics and Automation (ICRA)*, 26-30 May 2015, Seattle WA USA, pp. 2042–2047, ISBN 978-1-4799-6923-4, doi:10.1109/ICRA.2015.7139466
- Larsen, Kim et al. (2017) Larsen, Lars, Kim, Jonghwa, Kupke, Michael, and Schuster, Alfons, 2017. Automatic Path Planning of Industrial Robots Comparing Sampling-based and Computational Intelligence Methods, *Procedia Manufacturing* **Vol. 11**(C) pp. 241–248, doi:10.1016/j.promfg.2017.07.237
- Latombe (1991) Latombe, Jean-Claude, 1991. *Robot Motion Planning*, New York: Springer Science+Business Media, ISBN 978-0-7923-9206-9
- Laumond, Mansard et al. (2014) Laumond, Jean-Paul, Mansard, Nicolas, and Lasserre, Jean-Bernard, 2014. Optimality in Robot Motion: Optimal Versus Optimized Motion, *Communications of the ACM* **Vol. 57**(9) pp. 82–89, doi:10.1145/2629535
- Lavalle (1998) Lavalle, Steven M., 1998. Rapidly-Exploring Random Trees: A New Tool for Path Planning, URL <http://msl.cs.illinois.edu/~lavalle/papers/Lav98c.pdf>, retrieved: 22.03.18
- Lavalle (2003) Lavalle, Steven M., 2003. Motion Strategy Library, URL <http://msl.cs.uiuc.edu/msl/>, retrieved: 22.03.18
- Lavalle (2006) Lavalle, Steven M., 2006. *Planning Algorithms*, Cambridge U.K.: Cambridge University Press, ISBN 9781139455176
- Lavalle (2011) Lavalle, Steven M., 2011. Motion Planning: Wild Frontiers, *IEEE Robotics and Automation Society Magazine* **Vol. 18**(2) pp. 108–118, doi:10.1109/MRA.2011.941635
- Lavalle, Branicky et al. (2004) Lavalle, Steven M., Branicky, Michael S., and Lindemann, Stephen R., 2004. On the Relationship between Classical Grid Search and Probabilistic Roadmaps, *International Journal of Robotics Research* **Vol. 23**(7-8) pp. 673–692, doi:10.1177/0278364904045481
- Leali, Pellicciari et al. (2013) Leali, Francesco, Pellicciari, Marcello, Pini, Fabio, Berselli, Giovanni, and Vergnano, Alberto, 2013. An Offline Programming Method for the Robotic Deburring of Aerospace Components, In: Neto, Pedro and Moreira, António Paulo (Eds.), *Robotics in Smart Manufacturing: International Workshop, WRSM 2013, Co-located with*

-
- FAIM 2013, Porto, Portugal, June 26-28, 2013. *Proceedings*, Berlin Heidelberg: Springer, pp. 1–13, ISBN 978-3-642-39223-8, doi:10.1007/978-3-642-39223-8_1
- Leali, Pini et al. (2013) Leali, Francesco, Pini, Fabio, and Ansaloni, Matteo, 2013. Integration of CAM off-line programming in robot high-accuracy machining, In: *Proceedings of the 2013 IEEE/SICE International Symposium on System Integration*, 15-17 December 2013, Kobe Japan, pp. 580–585, doi:10.1109/SII.2013.6776741
- Lee, Kang et al. (1999) Lee, Young Dae, Kang, Byung Hun, and Park, Jong Oh, 1999. Robotic deburring strategy using burr shape recognition, In: *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients*, 17-21 October 1999, Kyongju South Korea, Vol. 3, pp. 1513–1518, ISBN 0-7803-5184-3, doi:10.1109/IROS.1999.811693
- Lehmann, Halbauer et al. (2012) Lehmann, Christian, Halbauer, Marcel, Euhus, Dirk, and Overbeck, Daniel, 2012. Milling with industrial robots: Strategies to reduce and compensate process force induced accuracy influences, In: *Proceedings of the 17th IEEE International Conference on Emerging Technologies & Factory Automation*, 17-21 September 2012, Krakow Poland, pp. 1075–1078, doi:10.1109/ETFFA.2012.6489741
- Lehmann, Halbauer et al. (2013) Lehmann, Christian, Halbauer, Marcel, Zwaag, Johnny van der, Schneider, Ulrich, and Berger, Ulrich, 2013. Offline Path Compensation to Improve Accuracy of Industrial Robots for Machining Applications, In: *Branchentreff der Mess- und Automatisierungstechnik*, Düsseldorf: VDI-Verlag, pp. 147–152, ISBN 978-3-18-092209-6
- Leo and Selvaraj (2014) Leo, Princely F. and Selvaraj, T., 2014. Vision Assisted Robotic Deburring of Edge Burrs in Cast Parts, *Procedia Engineering* **Vol. 97** pp. 1906–1914, doi:10.1016/j.proeng.2014.12.344
- Liao, Wang et al. (2010) Liao, Xiaoping, Wang, Weidong, Lin, Yizhong, and Gong, Changliang, 2010. Time-optimal trajectory planning for a 6R jointed welding robot using adaptive genetic algorithms, In: *International Conference on Computer, Mechatronics, Control and Electronic Engineering*, 24-26 August 2010,

-
- Changchun China, Vol. 2, pp. 600–603,
doi:10.1109/CMCE.2010.5610401
- Lima and Bracarense (2009) Lima, Eduardo José and Bracarense, Alexandre Queiroz, 2009. Trajectory generation in robotic shielded metal arc welding during execution time, *Industrial Robot: An International Journal* **Vol. 36**(1) pp. 19–26,
doi:10.1108/01439910910924639
- Liu, Cheng et al. (2001) Liu, Y., Cheng, T., and Zuo, L., 2001. Adaptive Control Constraint of Machining Processes, *International Journal of Advanced Manufacturing Technology* **Vol. 17**(10) pp. 720–726, doi:10.1007/s001700170117
- Lopes and Pires (2011) Lopes, António M. and Pires, E. J. Solteiro, 2011. Optimization of the Workpiece Location in a Machining Robotic Cell, *International Journal of Advanced Robotic Systems* **Vol. 8**(6) pp. 37–46, doi:10.5772/45681
- Lozano-Perez (1983) Lozano-Perez, Tomás, 1983. Spatial Planning: A Configuration Space Approach, *IEEE Transactions on Computers* **Vol. C-32**(2) pp. 108–120,
doi:10.1109/TC.1983.1676196
- Lozano-Perez and Wesley (1979) Lozano-Perez, Tomás and Wesley, Michael A., 1979. An Algorithm for Planning Collision-free Paths Among Polyhedral Obstacles, *Communications of the ACM* **Vol. 22**(10) pp. 560–570, doi:10.1145/359156.359164
- Lozano-Perez, Mason et al. (1984) Lozano-Perez, Tomás, Mason, Matthew T., and Taylor, Russell H., 1984. Automatic Synthesis of Fine-Motion Strategies for Robots, *International Journal of Robotics Research* **Vol. 3**(1) pp. 3–24,
doi:10.1177/027836498400300101
- Lynch and Park (2017) Lynch, Kevin M. and Park, Frank C., 2017. *Modern Robotics: Mechanics, Planning, and Control*, New York USA: Cambridge University Press, 1. Edn., ISBN 9781107156300
- Maeda, Ushioda et al. (2008) Maeda, Yusuke, Ushioda, Tatsuya, and Makita, Satoshi, 2008. Easy robot programming for industrial manipulators by manual volume sweeping, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 19-23 May 2008, Pasadena CA USA, pp. 2234–2239,
doi:10.1109/ROBOT.2008.4543546

-
- Makhanov, Batanov et al. (2002) Makhanov, Stanislav S., Batanov, Dentcho, Bohez, Erik, Sonthipaumpoon, Kawin, Anotaipaiboon, Weerachai, and Tabucanon, Mario, 2002. On the tool-path optimization of a milling robot, *Computers & Industrial Engineering* **Vol. 43**(3) pp. 455–472, doi:10.1016/S0360-8352(02)00118-3
- Marble and Kostas (2017) Marble, James D. and Kostas, Bekris E., 2017. Asymptotically Near-Optimal Is Good Enough for Motion Planning, In: Christensen, Henrik and Oussama, Khatib (Eds.), *Robotics Research: The 15th International Symposium ISRR*, Cham: Springer International Publishing, pp. 419–436, ISBN 978-3-319-29363-9, doi:10.1007/978-3-319-29363-9_24
- Mark, Bohlin et al. (2014) Mark, Andreas, Bohlin, Robert, Segerdahl, Daniel, Edelvik, Fredrik, and Carlson, Johan S., 2014. Optimisation of robotised sealing stations in paint shops by process simulation and automatic path planning, *International Journal of Manufacturing Research* **Vol. 9**(1) pp. 4–26, doi:10.1504/IJMR.2014.059597
- Marvel, Messina et al. (2015) Marvel, Jeremy A., Messina, Elena R., Antonishek, Brian, Wyk, Karl Van, and Fronczek, Lisa J., 2015. Tools for Robotics in SME Workcells: Challenges and Approaches for Calibration and Registration, URL <http://dx.doi.org/10.6028/NIST.IR.8093>, retrieved: 22.03.18
- Mason (1981) Mason, Matthew T., 1981. Compliance and Force Control for Computer-Controlled Manipulators, *IEEE Transaction on Systems, Man, and Cybernetics* **Vol. 11**(6) pp. 418–432, doi:10.1109/TSMC.1981.4308708
- Matsas and Vosniakos (2010) Matsas, Elias and Vosniakos, George Christopher, 2010. Improving feasibility of robotic milling through robot placement optimisation, *Robotics and Computer-Integrated Manufacturing* **Vol. 26**(5) pp. 517–525, doi:10.1016/j.rcim.2010.04.001
- Matsuoka, Shimizu et al. (1999) Matsuoka, Shin-ichi, Shimizu, Kazunori, Yamazaki, Nobuyuki, and Oki, Yoshinari, 1999. High-speed end milling of an articulated robot and its characteristics, *Journal of Materials Processing Technology* **Vol. 95**(1–3) pp. 83–89, doi:10.1016/S0924-0136(99)00315-5

-
- McDermott, Ghallab et al. (1998) McDermott, Drew, Ghallab, Malik, Howe, Adele, Knoblock, Craig, Ram, Ashwin, Veloso, Manuela, Weld, Daniel, and Wilkins, David, 1998. PDDL the planning domain definition language: Version 1.2, URL <http://homepages.inf.ed.ac.uk/mfourman/tools/propplan/pddl.pdf>, retrieved: 22.03.18
- Meyer (2011) Meyer, Christian, 2011. *Aufnahme und Nachbearbeitung von Bahnen bei der Programmierung durch Vormachen von Industrierobotern*, Heimsheim: Jost Jetter Verlag, ISBN 978-3-939890-75-1, Stuttgart, Univ., Diss., 2011
- Minhas, Lehmann et al. (2011) Minhas, Sarfraz Ul-Haque, Lehmann, Christian, Städter, Jost Philipp, and Berger, Ulrich, 2011. Reconfigurable Strategies for Manufacturing Setups to confront Mass Customization Challenges, In: *21st International Conference on Production Research (ICPR 21)*, Stuttgart Germany: Fraunhofer Verlag, ISBN 3-8396-0293-9
- Mitsi, Bouzakis et al. (2008) Mitsi, S., Bouzakis, K.-D., Sagris, D., and Mansour, G., 2008. Robot path planning optimization, free of collisions, using a hybrid algorithm, In: *3rd International Conference on Manufacturing Engineering (ICMEN)*, 1-3 October 2008, Chalkidiki Greece, pp. 475–484
- Mohammad, Babariya et al. (2008) Mohammad, Mohammad, Babariya, Vipul, and Sobh, Tarek M., 2008. Modeling A Deburring Process, Using DELMIA V5®, *Journal of Engineering and Applied Sciences* **Vol. 3**(11) pp. 835–847
- Mousavi, Gagnol et al. (2017) Mousavi, Said, Gagnol, Vincent, Bouzgarrou, Belhassen C., and Ray, Pascal, 2017. Stability optimization in robotic milling through the control of functional redundancies, *Robotics and Computer-Integrated Manufacturing* **Vol. 50** pp. 181–192, doi:10.1016/j.rcim.2017.09.004
- Nelaturi, Burton et al. (2015) Nelaturi, Saigopal, Burton, Gregory, Fritz, Christian, and Kurtoglu, Tolga, 2015. Automatic spatial planning for machining operations, In: *IEEE International Conference on Automation Science and Engineering (CASE)*, 24-28 August 2015, Gothenburg Sweden, pp. 677–682, doi:10.1109/CoASE.2015.7294158
- Neto and Mendes (2013) Neto, Pedro and Mendes, Nuno, 2013. Direct off-line robot programming via a common CAD package, *Robotics and*

Autonomous Systems **Vol. 61**(8) pp. 896–910,
doi:10.1016/j.robot.2013.02.005

Neto, Mendes et al. (2013)

Neto, Pedro, Mendes, Nuno, Araújo, Ricardo, Pires, J. Norberto, and Moreira, António Paulo, 2013. High-level robot programming based on CAD: dealing with unpredictable environments, *Computing Research Repository (CoRR)* URL <https://arxiv.org/abs/1309.2086>, retrieved: 22.03.18

Nielsen (2003)

Nielsen, Johan, 2003. *Information Modeling of Manufacturing Processes: Information Requirements for Process Planning in a Concurrent Engineering Environment*, Stockholm Sweden, Royal Institute of Technology, Diss., 2003

Nilsson, Johansson et al. (2005)

Nilsson, Klas, Johansson, Rolf, Robertsson, Anders, Bischoff, Rainer, Brogårdh, Torgny, and Hägele, Martin, 2005. Productive Robots and the SMERobot Project, In: *Book of Abstracts of Third Swedish Workshop on Autonomous Robotics*, Lund: FOI

Owen, Croft et al. (2008)

Owen, William, Croft, Elizabeth, and Benhabib, Beno, 2008. Stiffness optimization for two-armed robotic sculpting, *Industrial Robot: An International Journal* **Vol. 35**(1) pp. 46–57, doi:10.1108/01439910810843289

Pan and Zhang (2008)

Pan, Zengxi and Zhang, Hui, 2008. Robotic machining from programming to process control: a complete solution by force control, *Industrial Robot: An International Journal* **Vol. 35**(5) pp. 400–409, doi:10.1108/01439910810893572

Pan, Polden et al. (2012)

Pan, Zengxi, Polden, Joseph, Larkin, Nathan, Van Duin, Stephen, and Norrish, John, 2012. Recent Progress on Programming Methods for Industrial Robots, *Robotics and Computer-Integrated Manufacturing* **Vol. 28**(2) pp. 87–94, doi:10.1016/j.rcim.2011.08.004

Pandremenos, Doukas et al. (2011)

Pandremenos, John, Doukas, Christos, Stavropoulos, Panagiotis, and Chrysosolouris, George, 2011. Machining With Robots: A Critical Review, In: *7th International Conference on Digital Enterprise Technology*, 28–30 September 2011, Athens Greece

Perzylo, Griffiths et al. (2015)

Perzylo, Alexander, Griffiths, Sascha, Lafrenz, Reinhard, and Knoll, Alois, 2015. Generating grammars for natural

-
- language understanding from knowledge about actions and objects, In: *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 6-9 December 2015, Zhuhai China, pp. 2008–2013, doi:10.1109/ROBIO.2015.7419068
- Perzylo, Somani et al. (2015) Perzylo, Alexander, Somani, Nikhil, Profanter, Stefan, Rickert, Markus, and Knoll, Alois, 2015. Toward efficient robot teach-in and semantic process descriptions for small lot sizes, In: *Proceedings of Robotics: Science and Systems (RSS), Workshop on Combining AI Reasoning and Cognitive Science with Robotics*, July 2015, Rome Italy, pp. 1–7
- Perzylo, Somani et al. (2016) Perzylo, Alexander, Somani, Nikhil, Profanter, Stefan, Kessler, Ingmar, Rickert, Markus, and Knoll, Alois, 2016. Intuitive instruction of industrial robots: Semantic process descriptions for small lot production, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 9-14 October 2016, Daejeon South Korea, pp. 2293–2300, doi:10.1109/IROS.2016.7759358
- Pfrommer, Schleipen et al. (2013) Pfrommer, Julius, Schleipen, Miriam, and Beyerer, Jürgen, 2013. PPRS: Production skills and their relation to product, process, and resource, In: *IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)*, 10-13 September 2013, Cagliari Italy, pp. 1–4, doi:10.1109/ETFA.2013.6648114
- Pfrommer, Stogl et al. (2014) Pfrommer, Julius, Stogl, Denis, Aleksandrov, Kiril, Schubert, Viktor, and Hein, Björn, 2014. Modelling and orchestration of service-based manufacturing systems via skills, In: *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, 16-19 September 2014, Barcelona Spain, pp. 1–4, ISBN 978-1-4799-4845-1, doi:10.1109/ETFA.2014.7005285
- Pfrommer, Stogl et al. (2015) Pfrommer, Julius, Stogl, Denis, Aleksandrov, Kiril, Escaida, Stefan Navarro, Hein, Björn, and Beyerer, Jürgen, 2015. Plug & produce by modelling skills and service-oriented orchestration of reconfigurable manufacturing systems, *at - Automatisierungstechnik* **Vol. 63** pp. 790–800, doi:10.1515/auto-2014-1157
- Pires, Machado et al. (2014) Pires, E. J. Solteiro, Machado, J. A. Tenreiro, Lopes, M. António, and Oliveira, P. B. de Moura, 2014. Optimal Location of the Workpiece in a PKM-Based Machining

-
- Robotic Cell: Chapter 73, In: *Robotics: Concepts, Methodologies, Tools, and Applications*, pp. 1500–1515, ISBN 1466646071, doi:10.4018/978-1-4666-4607-0.ch073
- Pires (2006) Pires, Norberto J., 2006. Robotics for small and medium enterprises: control and programming challenges, *Industrial Robot* **Vol. 33**(6)
- Pires (2009) Pires, Norberto J., 2009. New challenges for industrial robotic cell programming, *Industrial Robot: An International Journal* **Vol. 36**(1), doi:10.1108/ir.2009.04936aaa.002
- Pires, Godinho et al. (2004) Pires, Norberto J., Godinho, T., and Ferreira, P., 2004. CAD interface for automatic robot welding programming, *Industrial Robot: Vol. 31*(1) pp. 71–76, doi:10.1108/01439910410512028
- Pires, Veiga et al. (2009) Pires, Norberto J., Veiga, Germano, and Araújo, Ricardo, 2009. Programming-by-demonstration in the coworker scenario for SMEs, *Industrial Robot: An International Journal* **Vol. 36**(1) pp. 73–83, doi:10.1108/01439910910924693
- Polden (2014) Polden, Joseph, 2014. *Automated offline programming for low volume robotic manufacturing*, Wollongong Australia, Wollongong Univ., Diss., 2014
- Polden, Pan et al. (2011) Polden, Joseph, Pan, Zengxi, Larkin, Nathan, Van Duin, Stephen, and Norrish, John, 2011. Offline Programming for a Complex Welding System Using DELMIA Automation, In: Tarn, Tzyh, Jong, Chen, Shan, Ben, and Fang, Gu (Eds.), *Robotic Welding, Intelligence and Automation: RWIA'2010*, Berlin Heidelberg: Springer, pp. 341–349, ISBN 978-3-642-19959-2, doi:10.1007/978-3-642-19959-2_42
- Polden, Pan et al. (2017) Polden, Joseph, Pan, Zengxi, Larkin, Nathan, and Van Duin, Stephen, 2017. Adaptive Partial Shortcuts: Path Optimization for Industrial Robotics, *Journal of Intelligent & Robotic Systems* **Vol. 86**(1) pp. 35–47, doi:10.1007/s10846-016-0437-x
- Porta, Ros et al. (2014) Porta, Josep M., Ros, Lluís, Bohigas, Oriol, Manubens, Montserrat, Rosales, Carlos, and Jaillet, Leonard, 2014. The CUIK Suite: Analyzing the Motion Closed-Chain Multibody

-
- Systems, *IEEE Robotics Automation Magazine* **Vol. 21**(3) pp. 105–114, doi:10.1109/MRA.2013.2287462
- Rickert and Gaschler (2017) Rickert, Markus and Gaschler, Andre, 2017. Robotics Library: An Object-Oriented Approach to Robot Applications, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 24-28 September 2017, Vancouver Canada, doi:10.1109/IROS.2017.8202232
- Rutgeerts (2007) Rutgeerts, Johan, 2007. *Task Specification and Estimation for Sensor-Based Robot Tasks in the Presence of Geometric Uncertainty*, Leuven, Katholieke Leuven Univ., Diss., 2007
- Sang, Tae et al. (2010) Sang, Su Choi, Tae, Hyuck Yoon, and Sang, Do Noh, 2010. XML-based neutral file and PLM integrator for PPR information exchange between heterogeneous PLM systems, *Int. J. Computer Integrated Manufacturing* pp. 216–228, doi:10.1080/09511920903443234
- Schleipen and Drath (2009) Schleipen, Miriam and Drath, Rainer, 2009. Three-view-concept for Modeling Process or Manufacturing Plants with AutomationML, *International Conference on Emerging Technologies and Factory Automation (ETFA)* pp. 1–4, doi:10.1109/ETFA.2009.5347260
- Schleipen, Pfrommer et al. (2014) Schleipen, Miriam, Pfrommer, Julius, Stogl, Denis, Hein, Björn, Aleksandrov, Kiril, Aleksandrov, Stefan, and Aleksandrov, Jürgen, 2014. AutomationML to describe skills of production plants based on the PPR concept, In: *3rd AutomationML User Conference*, 7-8 October 2014, Blomberg Germany
- Schmirgel, Zimmermann et al. (2007) Schmirgel, Volker, Zimmermann, Uwe E., Hulin, Thomas, and Preusche, Carsten, 2007. Position Paper: Human Skills for Programming-by-Demonstration of Robots, URL <http://elib.dlr.de/55618/>, retrieved: 22.03.18
- Schneider (2017) Schneider, Ulrich, 2017. *Untersuchung von modellbasierter Fehlerkompensation und erweiterter Positionsregelung zur Genauigkeitssteigerung von roboterbasierten Zerspanungsprozessen*, Stuttgart Germany: Fraunhofer Verlag, ISBN 978-3-8396-1123-4, Stuttgart, Univ., Diss., 2016
- Schneider, Ansaloni et al. (2013) Schneider, Ulrich, Ansaloni, Matteo, Drust, Manuel, Leali, Francesco, and Verl, Alexander, 2013. Experimental

Investigation of Sources of Error in Robot Machining, In: Neto, Pedro and Moreira, António Paulo (Eds.), *Robotics in Smart Manufacturing*, Berlin Heidelberg: Springer, Communications in Computer and Information Science, Vol. 371, pp. 14–26, ISBN 978-3-642-39222-1, doi:10.1007/978-3-642-39223-8_2

- Schneider, Diaz Posada et al. (2013) Schneider, Ulrich, Diaz Posada, Julian Ricardo, Drust, Manuel, and Verl, Alexander, 2013. Position control of an industrial robot using an optical measurement system for machining purposes, In: *11th International Conference on Manufacturing Advanced manufacturing technology XXVII.*, 19-20 September 2013, Cranfield UK, pp. 307–312, ISBN 978-1-907413-23-0
- Schneider, Diaz Posada et al. (2014) Schneider, Ulrich, Diaz Posada, Julian Ricardo, Drust, Manuel, Verl, Alexander, and Zwaag, Johnny van der, 2014. Combining holistic programming with kinematic parameter optimization for robot machining, In: *Proceedings of 41st International Symposium on Robotics ISR/Robotik*, 2-3 June 2014, Munich Germany, pp. 1–7, ISBN 978-3-8007-3601-0
- Schneider, Drust et al. (2014) Schneider, Ulrich, Drust, Manuel, Ansaloni, Matteo, Lehmann, Christian, Pellicciari, Marcello, Leali, Francesco, Gunnink, Jan Willem, and Verl, Alexander, 2014. Improving robotic machining accuracy through experimental error investigation and modular compensation, *International Journal of Advanced Manufacturing Technology* pp. 1–13, doi:10.1007/s00170-014-6021-2
- Schneider, Momeni-K et al. (2014) Schneider, Ulrich, Momeni-K, Mahdi, Ansaloni, Mateo, and Verl, Alexander, 2014. Stiffness modeling of industrial robots for deformation compensation in machining, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 14-18 September 2014, Chicago IL USA, pp. 4464–4469, doi:10.1109/IROS.2014.6943194
- Schneider, Diaz Posada et al. (2015) Schneider, Ulrich, Diaz Posada, Julian Ricardo, and Verl, Alexander, 2015. Automatic Pose Optimization for Robotic Processes, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 26-30 May 2015, Seattle WA USA, pp. 2054–2059, ISBN 978-1-4799-6923-4, doi:10.1109/ICRA.2015.7139468

-
- Schraft and Hartfuss (1994) Schraft, Rolf Dieter and Hartfuss, Christoph, 1994. Knowledge-based programming of industrial robots for arc welding of small lot sizes, In: International Federation of robotics (Ed.), *International Symposium on Industrial Robots*, 25-27 April 1994, Hannover, pp. 427–434
- Schraft and Meyer (2006) Schraft, Rolf Dieter and Meyer, Christian, 2006. The Need For An Intuitive Teaching Method For Small And Medium Enterprises, In: *Proceedings of the Joint Conference on Robotics. ISR 2006*, Düsseldorf: VDI-Verlag, ISBN 3-18-091956-6
- Scioni, Borghesan et al. (2015) Scioni, Enea, Borghesan, Gianni, Bruyninckx, Herman, and Bonfè, Marcello, 2015. Bridging the gap between discrete symbolic planning and optimization-based robot control, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 26-30 May 2015, Seattle WA USA, pp. 5075–5081, ISBN 978-1-4799-6923-4, doi:10.1109/ICRA.2015.7139905
- Short, Pan et al. (2016) Short, Andrew, Pan, Zengxi, Larkin, Nathan, and Duin, Stephen van, 2016. Recent progress on sampling based dynamic motion planning algorithms, In: *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 12-15 July 2016, Banff Canada, pp. 1305–1311, doi:10.1109/AIM.2016.7576950
- Siciliano, Sciavicco et al. (2010) Siciliano, Bruno, Sciavicco, Lorenzo, Villani, Luigi, and Oriolo, Giuseppe, 2010. *Robotics: Modelling, Planning and Control*, London: Springer, ISBN 978-1-84628-641-4
- Slamani and Bonev (2013) Slamani, Mohamed and Bonev, Ilian A., 2013. Characterization and experimental evaluation of gear transmission errors in an industrial robot, *Industrial Robot: An International Journal* **Vol. 40**(5) pp. 441–449, doi:10.1108/IR-07-2012-387
- Slavkovic, Milutinovic et al. (2014) Slavkovic, Nikola R., Milutinovic, Dragan S., and Glavonjic, Milos M., 2014. A method for off-line compensation of cutting force-induced errors in robotic machining by tool path modification, *International Journal of Advanced Manufacturing Technology* **Vol. 70**(9-12) pp. 2083–2096, doi:10.1007/s00170-013-5421-z
- Song and Song (2013) Song, Hee-Chan and Song, Jae-Bok, 2013. Precision robotic deburring based on force control for arbitrarily shaped

-
- workpiece using CAD model matching, *International Journal of Precision Engineering and Manufacturing* **Vol. 14**(1) pp. 85–91, doi:10.1007/s12541-013-0013-2
- Sörnmo, Olofsson et al. (2012a) Sörnmo, Olof, Olofsson, Björn, Robertsson, Anders, and Johansson, Rolf, 2012. Increasing Time-Efficiency and Accuracy of Robotic Machining Processes Using Model-Based Adaptive Force Control, In: *10th IFAC Symposium on Robot Control*, 5-7 September 2012, Dubrovnik Croatia, pp. 543–548, doi:10.3182/20120905-3-HR-2030.00065
- Sörnmo, Olofsson et al. (2012b) Sörnmo, Olof, Olofsson, Björn, Schneider, Ulrich, Robertsson, Anders, and Johansson, Rolf, 2012. Increasing the Milling Accuracy for Industrial Robots Using a Piezo-Actuated High-Dynamic Micro Manipulator, In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, 11-14 July 2012, Kachsiung Taiwan, pp. 104–110, ISBN 978-1-4673-2575-2, doi:10.1109/AIM.2012.6265942
- Spensieri, Carlson et al. (2016a) Spensieri, Domenico, Carlson, Johan S., Bohlin, Robert, Kressin, Jonas, and Shi, Jane, 2016. Optimal Robot Placement for Tasks Execution, *Procedia CIRP* **Vol. 44** pp. 395–400, doi:10.1016/j.procir.2016.02.105
- Spensieri, Carlson et al. (2016b) Spensieri, Domenico, Carlson, Johan S., Ekstedt, Fredrik, and Bohlin, Robert, 2016. An Iterative Approach for Collision Free Routing and Scheduling in Multirobot Stations, *IEEE Transactions on Automation Science and Engineering* **Vol. 13**(2) pp. 950–962, doi:10.1109/TASE.2015.2432746
- Stanford Univ. (2006) Stanford Univ., 2006. MPK - Motion Planning Kit, URL <http://robotics.stanford.edu/~mitul/mpk/>, retrieved: 22.03.18
- Staranowicz and Mariottini (2011) Staranowicz, Aaron and Mariottini, Gian Luca, 2011. A Survey and Comparison of Commercial and Open-source Robotic Simulator Software, In: *Proceedings of the 4th International Conference on Pervasive Technologies Related to Assistive Environments*, New York: ACM, pp. 1–8, ISBN 978-1-4503-0772-7, doi:10.1145/2141622.2141689
- Starek, Gómez et al. (2015) Starek, Joseph A., Gómez, Javier V., Schmerling, Edward, Janson, Lucas, Moreno, Luis, and Pavone, Marco, 2015. An

-
- Asymptotically-Optimal Sampling-Based Algorithm for Bi-directional Motion Planning, *Computing Research Repository (CoRR)* doi:10.1109/IROS.2015.7353652
- Stenmark (2015) Stenmark, Maj, 2015. *Instructing Industrial Robots Using High-Level Task Descriptions*, Lund Sweden, Lund Univ., Diss., 2015
- Stenmark and Malec (2015) Stenmark, Maj and Malec, Jacek, 2015. Knowledge-based Instruction of Manipulation Tasks for Industrial Robotics, *Robotics and Computer-Integrated Manufacturing* **Vol. 33**(C) pp. 56–67, doi:10.1016/j.rcim.2014.07.004
- Stenmark and Nugues (2013) Stenmark, Maj and Nugues, Pierre, 2013. Natural language programming of industrial robots, In: *44th International Symposium on Robotics (ISR)*, 24-26 Oct 2013, Seoul South Korea, pp. 1–5, doi:10.1109/ISR.2013.6695630
- Stilman (2007) Stilman, Mike, 2007. Task constrained motion planning in robot joint space, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 29.10.-2.11.2007, San Diego CA USA, pp. 3074–3081, ISBN 978-1-4244-0911-2, doi:10.1109/IROS.2007.4399305
- Stilman (2010) Stilman, Mike, 2010. Global Manipulation Planning in Robot Joint Space With Task Constraints, *IEEE Transactions on Robotics* **Vol. 26**(3) pp. 576–584, doi:10.1109/TRO.2010.2044949
- Subrin, Sabourin et al. (2013) Subrin, Kévin, Sabourin, Laurent, Cousturier, Richard, Gogu, Grigore, and Mezouar, Youcef, 2013. New Redundant Architectures in Machining: Serial and Parallel Robots, *Procedia Engineering* **Vol. 63**(C) pp. 158–166, doi:10.1016/j.proeng.2013.08.203
- Şucan, Moll et al. (2012) Şucan, Ioan A., Moll, Mark, and Kavraki, Lydia E., 2012. The Open Motion Planning Library, *IEEE Robotics & Automation Magazine* **Vol. 19**(4) pp. 72–82, doi:10.1109/MRA.2012.2205651
- Tang, Khaksar et al. (2012) Tang, Sai Hong, Khaksar, W., Ismail, N. B., and Ariffin, M. K. A., 2012. A review on robot motion planning approaches, *Pertanika Journal of Science and Technology* **Vol. 20**(1) pp. 15–29
- Thomas, Barrenscheen et al. (2003) Thomas, Ulrike, Barrenscheen, Mark, and Wahl, Friedrich M., 2003. Efficient assembly sequence planning

-
- using stereographical projections of C-space obstacles, In: *Proceedings of the IEEE International Symposium on Assembly and Task Planning*, 10-11 July 2003, Besancon France, pp. 96–102, doi:10.1109/ISATP.2003.1217194
- Thomas, Hirzinger et al. (2013) Thomas, Ulrike, Hirzinger, Gerd, Rumpe, Bernhard, Schulze, Christoph, and Wortmann, Andreas, 2013. A new skill based robot programming language using UML/P Statecharts, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 6-10 May 2013, Karlsruhe Germany, pp. 461–466, doi:10.1109/ICRA.2013.6630615
- Ting, Lei et al. (2002) Ting, Y., Lei, W. I., and Jar, H. C., 2002. A Path Planning Algorithm for Industrial Robots, *Computers & Industrial Engineering* **Vol. 42**(2-4) pp. 299–308, doi:10.1016/S0360-8352(02)00013-X
- Tom, Yuval et al. (2015) Tom, Erez, Yuval, Tassa, and Emanuel, Todorov, 2015. Simulation tools for model-based robotics: Comparison of Bullet, Havok, MuJoCo, ODE and PhysX, In: *IEEE International Conference on Robotics and Automation (ICRA)*, 26-30 May 2015, Seattle WA USA, pp. 4397–4404, ISBN 978-1-4799-6923-4, doi:10.1109/ICRA.2015.7139807
- Toussaint, Ritter et al. (2015) Toussaint, Marc, Ritter, Helge, and Brock, Oliver, 2015. The Optimization Route to Robotics and Alternatives, *KI - Künstliche Intelligenz* **Vol. 29**(4) pp. 379–388, doi:10.1007/s13218-015-0379-7
- Tsianos, Şucan et al. (2007) Tsianos, Konstantinos I., Şucan, Ioan A., and Kavraki, Lydia E., 2007. Sampling-Based Robot Motion Planning: Towards Realistic Applications, *Computer Science Review* **Vol. 1**(1) pp. 2–11
- Tyapin, Kaldestad et al. (2016) Tyapin, Ilya, Kaldestad, Knut B., and Hovland, Geir, 2016. Off-line path correction of robotic face milling using static tool force and robot stiffness, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 9-14 October 2016, Daejeon South Korea, pp. 5506–5511, doi:10.1109/IROS.2015.7354157
- Verbarg (1997) Verbarg, Knut, 1997. Planning of Fast and Secure Paths in Complex Configuration Spaces Using Simple Primitives, In: *Intelligent Robots: Sensing, Modeling and Planning*, 1-6 September 1997, Dagstuhl Germany: World Scientific Press, pp. 18–32, ISBN 9810231857

-
- Verbarg (1998) Verbarg, Knut, 1998. Motion-planning for welding robots, In: *Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (IECON)*, 31.08.-4.09.1998, Aachen Germany, pp. 2227–2232, doi:10.1109/IECON.1998.724067
- Vosniakos and Chronopoulos (2009) Vosniakos, G. C. and Chronopoulos, A., 2009. Industrial robot path planning in a constraint-based computer-aided design and kinematic analysis environment, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* **Vol. 223**(5) pp. 523–533, doi:10.1243/09544054JEM1234
- Vuong, Lim et al. (2013) Vuong, N. D., Lim, T. M., and Yang, G., 2013. Simulation and Off-Line Programming for Contact Operations, In: Nee, Andrew (Ed.), *Handbook of Manufacturing Engineering and Technology*, London: Springer, pp. 1–18, ISBN 978-1-4471-4976-7, doi:10.1007/978-1-4471-4976-7_98-1
- Wang, Li et al. (2011) Wang, HaiPeng, Li, Nuo, Fang, Xin Yu, and Hofschulte, Jens, 2011. An architecture for robot application simulation systems and its industry practice, In: *IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM)*, 17-19 September 2011, Qingdao China, pp. 143–147, doi:10.1109/RAMECH.2011.6070471
- Wang, Zhang et al. (2009) Wang, J., Zhang, H., and Fuhlbrigge, T., 2009. Improving machining accuracy with robot deformation compensation, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 10-15 October 2009, St. Louis MO USA, pp. 3826–3831, doi:10.1109/IROS.2009.5353988
- Wang, Shi et al. (2015) Wang, Xue-Wu, Shi, Ying-Pan, Gu, Xing-Sheng, and Ding, Dong-Yan, 2015. Partition Mutation PSO for Welding Robot Path Optimization, In: Tarn, Tzyh-Jong, Chen, Shan-Ben, and Chen, Xiao-Qi (Eds.), *Robotic Welding, Intelligence and Automation: RWIA'2014*, Cham: Springer International Publishing, pp. 77–86, ISBN 978-3-319-18997-0, doi:10.1007/978-3-319-18997-0_6
- Wang, Shi et al. (2016) Wang, Xuewu, Shi, Yingpan, Ding, Dongyan, and Gu, Xingsheng, 2016. Double global optimum genetic algorithm particle swarm optimization-based welding robot path planning, *Engineering Optimization* **Vol. 48**(2) pp. 299–316, doi:10.1080/0305215X.2015.1005084

-
- Canudas de Wit, Bastin et al. (1996) Canudas de Wit, Carlos, Bastin, Georges, and Siciliano, Bruno (Eds.), 1996. *Theory of Robot Control*, London: Springer-Verlag, ISBN 3540760547
- Wolfe, Marthi et al. (2010) Wolfe, Jason, Marthi, Bhaskara, and Russell, Stuart, 2010. Combined Task and Motion Planning for Mobile Manipulation, In: *Proceedings of the Twentieth International Conference on Automated Planning and Scheduling*, 12-16 May 2010, Toronto Canada: AAAI Press, pp. 254–257
- Xiao and Huan (2012) Xiao, Wenlei and Huan, Ji, 2012. Redundancy and optimization of a 6R robot for five-axis milling applications: singularity, joint limits and collision, *Production Engineering* **Vol. 6**(3) pp. 287–296, doi:10.1007/s11740-012-0362-1
- Xiong, Ding et al. (2017) Xiong, Gang, Ding, Ye, and Zhu, LiMin, 2017. A Feed-Direction Stiffness Based Trajectory Optimization Method for a Milling Robot, In: Huang, YongAn, Wu, Hao, Liu, Honghai, and Yin, Zhouping (Eds.), *Intelligent Robotics and Applications: 10th International Conference, ICRA 2017, Wuhan, China, August 16–18, 2017, Proceedings, Part II*, Cham: Springer International Publishing, pp. 184–195, ISBN 978-3-319-65292-4, doi:10.1007/978-3-319-65292-4_17
- Yang, Qi et al. (2016) Yang, Liang, Qi, Juntong, Song, Dalei, Xiao, Jizhong, Han, Jianda, and Xia, Yong, 2016. Survey of Robot 3D Path Planning Algorithms, *Journal of Control Science and Engineering* doi:10.1155/2016/7426913
- Yao and Gupta (2005) Yao, Zhenwang and Gupta, Kamal, 2005. Path planning with general end-effector constraints: using task space to guide configuration space search, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2-6 August 2005, Edmonton Alta Canada, pp. 1875–1880, doi:10.1109/IROS.2005.1545305
- Zhang, Chen et al. (2006) Zhang, Hui, Chen, Heping, Xi, Ning, Zhang, George, and He, Jianmin, 2006. On-Line Path Generation for Robotic Deburring of Cast Aluminum Wheels, In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 9-15 October 2006, Beijing China, pp. 2400–2405, doi:10.1109/IROS.2006.281679

Zhang, Cabaravdic et al. (2004)

Zhang, Xiang, Cabaravdic, Malik, Kneupner, Klaus, and Kuhlenkoetter, Bernd, 2004. Real-Time Simulation of Robot Controlled Belt Grinding Processes of Sculptured Surfaces, *International Journal of Advanced Robotic Systems* **Vol. 1**(2) pp. 109–114, doi:10.5772/5627

Zhu and Hu (2018)

Zhu, Zuyuan and Hu, Huosheng, 2018. Robot Learning from Demonstration in Robotic Assembly: A Survey, *Robotics* **Vol. 7**(2), doi:10.3390/robotics7020017, URL <http://www.mdpi.com/2218-6581/7/2/17>

Robotic programming is still a challenge in manufacturing. Reasons for this challenge are the time-consuming, complex, primarily manually optimized, and expert-dependent path generation. Therefore, easy and optimized programmable robotic manufacturing systems are required.

To make up for this deficiency, this dissertation raises the following research question: How to automatically generate offline optimized paths for robotic manufacturing processes (RMPs) under constraints and process optimization criteria? This research question is answered with the following evaluated thesis: Robotic paths can be automatically optimized by methodologically configuring and deploying sample-based generation algorithms, based on the procedural interpretation of the Product, Process, and Resource model-based components of the RMP.

The generalized approach and implemented architecture is named Automatic Optimized Offline Programming. Three different use-cases are presented for evaluating the path planning approach: (1) The optimized collision avoidance in robotic welding, (2) the optimized stiffness for robotic milling, and (3) the optimized sensing strategy for robotic deburring.

ISBN 978-3-8396-1614-7



FRAUNHOFER VERLAG