TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Montagetechnik und Betriebswissenschaften am Institut für Werkzeugmaschinen und Betriebswissenschaften (*iwb*)
der Technischen Universität München

A New Programming Approach for Robot-based Flexible Inspection Systems

Dipl.-Ing. (Univ.)

William Brice Tekouo Moutchiho

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr.-Ing. Michael Friedrich Zäh

Prüfer der Dissertation:

1. Univ.-Prof. Dr.-Ing. Gunther Reinhart
2. Univ.-Prof. Dr. rer. nat. Tim Christian Lüth

Die Dissertation wurde am 02.02.2012 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 25.04.2012 angenommen.

Editor's Preface

Production engineering is of central importance to the continuous development of our industrial society. The performance capacity of manufacturing companies is highly dependent on the utilized resources, the applied production techniques, and the implemented organization strategies. Guaranteeing a company's success entails an ideal combination of technology, organization and workforce management.

Achieving an optimal configuration of cost, time and quality is a highly complex and intricate task which requires production strategies to be continuously monitored, reviewed and enhanced. This involves both a reduction in, and a sound understanding of, the complexity of products, manufacturing systems and operations.

The *iwb* strives to continually improve production systems, planning processes and manufacturing technologies. In each of these aspects, particular focus is given to employee-oriented requirements. Despite an increase in the degree of automation, human employees play an important role in the product development process, and need be optimally integrated into the manufacturing process.

The research presented in this document is part of a series which spans through the various research areas of the *iwb*. These areas range from the development and planning of production systems as a whole, to more specific technologies such as individual manufacturing and assembly processes. Research in supply chain management and changeable production systems is aimed at optimizing production processes. Virtual production methods such as simulation tools are implemented at all levels of the production process, for testing and evaluation purposes. In terms of manufacturing technologies, the *iwb* conducts innovative research and develops new concepts in the areas of microassembly and handling, bonding processes, mechatronics, rapid manufacturing and laser technologies.

The *iwb* research series presents new results and findings which are highly relevant to the industrial world, and therefore serves as a transfer of knowledge between the university and industry.

Gunther Reinhart *Michael Zäh*

Acknowledgements

This dissertation is the result of my work at the Institute for Machine Tools and Industrial Management *(iwb)* at the Technische Universität München.

I owe my supervisor Prof. Dr.-Ing. Gunther Reinhart the deepest gratitude for his guidance, eye-opening discussions and crucial nudges in the right direction over the past five years. I am grateful to Prof. Dr.-Ing. Michael Zäh for his keen interest in me and my work, as well as the encouragement he gave me throughout. Not least, I am obliged to Prof. Dr. rer. nat. Tim Christian Lüth for his willingness to be part of the examination committee and for the time and effort he invested.

Cordial thanks go to my institute colleagues who have helped me in many matters during my time at the *iwb* especially Dipl.-Ing. Zitzmann, Dr. Pörnbacher, Dr. Vogl, Dr. Munzert, Dr. Radi, Dipl.-Ing. Braunreuther, my colleagues from the robotic and automation department as well as all the service centers' staff. I would also like to thank my former student Dipl.-Ing. Tschochner for his pertinent contribution to the success of this dissertation.

I wish to gratefully acknowledge the support of Perceptron Inc. in terms of high-tech equipment and industrial application examples.

There are many people to whom I am indebted for making my life a productive and enjoyable experience. First and foremost, my parents Emilienne Nguiedoum and Pierre Moutchiho, who have always shown me guidance, patience and love, and who untiringly reminded me of the importance of education. Although neither of you possessed a tertiary degree, you both worked so hard to give your children the best possible education. Profuse thanks go to my siblings Alain, Irene, Rodrigue, Leopold, Armand and Karlette for always being supportive of my endeavors. I am also deeply obliged to my uncle Mr. Foumbi Joseph for his immeasurable support for our family and for being a role model for me. Without this great family, I never would have been able to make it this far – thank you!

Words fail to describe my appreciation to Julia for her support, confidence in me, and unconditional love over the past years. Without you, my path would have certainly been more difficult – thank you!

Finally, I will be always indebted to Germaine for her confidence in me and for enabling me to move to Germany. This gratitude also extends to Delphine, Etienne, and Junang for their relentless support at my arrival in Germany – thank you!

Munich, April 2012 *William Brice Tekouo Moutchiho*

This work is dedicated to my Mother and my late father and brother Alain who didn't live long enough to see me reach this accomplishment.

Abstract

When a workpiece becomes defective at an early stage in the production process, unnecessary costs are incurred by completing the manufacture of a product that cannot be sold. Therefore, it is of fundamental interest to manufacturers that faulty workpieces are identified as soon as they are misprocessed. To this end, flexible metrology equipment, such as flexible inspection systems (FISs), are deployed across manufacturing lines to locally assess the quality conformity of workpieces before they are released to the next production steps. In their most versatile configuration, FISs consist of a robot manipulator that carries a high-accuracy optical sensor on its tip. To carry out an inspection task, FISs operate by first capturing a 3D range image of the physical workpiece under investigation. This range image, also referred to as point cloud, is then compared with the workpiece's CAD model to detect potential geometrical deficiencies.

During the past few years, a large number of FISs have been deployed on manufacturing floors. However, the range of applications is still restricted to those in which FISs are used to repeatedly gauge a small number of predefined standard components. These days, FISs still have to find their way into small batch or high-variety manufacturing environments. The greatest hurdle that prevents them from doing so is the huge effort required to invest in programming activities to adapt them to different inspection tasks. This programming effort is necessitated by the myriad of restrictions that have to be simultaneously fulfilled during digitization operations by both the sensor and the manipulator. Existing programming methods and system still fall short of properly addressing the aforementioned challenges.

Consequently, the research described is this doctoral dissertation is geared towards developing a programming methodology that reduces the amount of effort currently necessary for programming FISs. The methodology is based on a task-oriented approach and consists of five steps: a) the extraction of feature contour lines from the CAD model of the workpiece, b) the description of the inspection task based on the extracted features lines, c) the automatic generation of sensor trajectories that allow the digitization of these features, d) the automatic synthetization of feasible movements of the manipulator carrying the sensor and, finally, e) the online adaptation of the sensor trajectories at execution time to cope with modeling inaccuracy issues related to model-based programming.

The key advantage of this new methodology is that operators are shielded from technical details and therefore do not need to familiarize themselves with either robotics or metrology. This approach is believed to take the flexibility of FISs up another notch by deskilling their programming processes. As a result, the utilization ratio of FISs is expected to considerably increase, opening up access to new market opportunities, especially in small batch manufacturing companies.

Zusammenfassung

Zur Überprüfung der Form- und Maßhaltigkeit von Blechbauteilen werden in zunehmendem Maße robotergestützte flexible Messsysteme eingesetzt. Eine wesentliche Herausforderung bei der Neu- und Umkonfiguration solcher Systeme besteht aktuell in dem unverhältnismäßig hohen Programmieraufwand. Dieser liegt darin begründet, dass die Trajektorie des optischen Messsensors und die Bewegung des Industrieroboters eine Vielzahl an komplexen Einschränkungen simultan erfüllen müssen.

Im Rahmen der vorliegenden Dissertation wurde daher eine neue Methodik zur effizienten und aufwandsarmen Programmierung von robotergestützten Messsystemen entworfen und prototypisch implementiert. Diese Methodik zeichnet sich aus durch eine CAD-basierte Beschreibung der Messaufgabe, eine automatisierte Generierung von optimalen Sensortrajektorien und Roboterbewegungen sowie eine Kompensation von Trajektorieabweichungen während der Realisierung der Messaufgabe. Desweiteren unterscheidet sich diese von existierenden Lösungen durch die Tatsache, dass weder tief greifende messtechnische Kenntnisse noch Roboter- oder IT-Kenntnisse seitens des Programmierers benötigt werden.

"Talent helps but it won't take you as far as ambition."

Paul Arden

Contents

Abbreviations

Abbreviations	Description
API	**A**pplication **P**rogramming **I**nterface
AoI	**A**reas **o**f **I**nterest
APS	**A**utomatic **P**rogramming **S**ystem
APoFIS	**A**utomatic **P**rogramming of **F**lexible **I**nspection **S**ystem
BIBO	**B**ounded **I**nput **B**ounded **O**utput
B-reps	**B**oundary **rep**resentations
CAD	**C**omputer **A**ided **D**esign
CAx	**C**omputer **A**ided methods
CIM	**C**omputer **I**ntegrated **M**anufacturing
CMM	**C**oordinate **M**easuring **M**achine
CPU	**C**entral **P**rocessing **U**nit
DLS	**D**amped **L**east **S**quare
DXF	**D**rawing Interchange File **F**ormat
DoF	**D**egree **o**f **F**reedom
DoFi	**D**epth-**o**f-**Fi**eld
e.g.	**E**xempli **G**ratia
FoV	**F**ield-**o**f-**V**iew
FIFO	**F**irst **I**n, **F**irst **O**ut
FIS	**F**lexible **I**nspection **S**ystem
GUI	**G**raphical **U**ser **I**nterface
IEC	**I**nternational **E**lectrotechnical **C**ommission
IGES	**I**nitial **G**raphics **E**xchange **S**pecification
ISO	**I**nternational **O**rganization for **S**tandardization
IR	**I**ndustry **R**obot
ISO	**I**nternational **O**rganization for **S**tandardization
iwb	**I**nstitut für **W**erkzeugmaschinen und **B**etriebswissenschaften

Abbreviations	Description
	(engl.: Institute for Machine Tools and Industrial Management)
LaWi	**L**aser **W**idth
MAT	**M**edial **A**xis **T**ransformation
MATLAB	**Mat**lab **Lab**oratory
PRM	**P**robabilistic **R**oadmap **M**ethods
PQP	**P**roximity **Q**uery **P**ackage
PTP	**P**oint-**T**o-**P**oint
QI	**Q**uality **I**nspection
STEP	**ST**andard for the **E**xchange of **P**roduct model data
STL	**S**urface **T**essellation **L**anguage
TCP	**T**ool **C**enter **P**oint
TCP/IP	**T**ransmission **C**ontrol **P**rotocol / **I**nternet **P**rotocol
ToP	**T**ask **o**riented **P**rogramming
WCP	**W**rist **C**enter **P**oint
WP	**W**ork**p**iece
3D	**T**hree **d**imensional

List of Symbols

Symbol	Unit	Description
θ_{lim}	°	Maximal incidence angle of the sensor
α_{LS}	°	Aperture angle of the laser plane
δ	°	Triangulation angle of the sensor
ε	-	Approximation error of the dichotomy algorithm
ξ	-	Damping factor of the closed loop system
Θ_k	°	Robot joint variables
$\Delta\Theta_k$	°	Difference between consecutive robot joint variables
λ_{max}	-	Maximum damping for the Damped Least Square
$\delta\theta_{1\ldots6}$	°	differential changes in robot joint coordinates
Δp	mm	Differential change in the robot TCP pose
$\delta x, \delta x, \delta x$	mm	Differential change in the coordinate x, y, z of the TCP pose
$\delta\alpha, \delta\beta, \delta\gamma$	°	Differential change of the coordinate α, β, γ
J	-	Jacobian matrix of the FIS
λ	-	Damping factor for the Damped Least Square
J^T	-	Transpose of the jacobian matrix
I	-	Identity matrix
$\sigma_{1\ldots6}$	-	Singular values of the jacobian matrix
τ	-	Size of the singular region of the FIS
$U\Sigma V^T$		Transpose of the singular value decomposition of the jacobian
$\hat{\sigma}$	-	Smallest singular value of the jacobian
G	-	Transfer function of the FIS
T	ms	Time constant
T_s	ms	Sampling time
z	-	Z-Transform
K_p	-	Gain factor of the closed-loop system
α, β, γ	°	ZYX-Euler angle set

Abbreviations

Symbol	Unit	Description
F_i	-	Frame attached to the *i*th link of the FIS
${}^{i}T_{i+1} =$	-	Transformation matrix from frame F_i to frame F_{i+1}
$\theta_i,$	°	DH-Parameter, rotation about the vector z_{i-1} of frame F_{i-1}
α_i	°	DH-Parameter, rotation about the vector x_i of frame F_i
d_i	mm	DH-Parameter, translation along the axis z_{i-1} of frame F_{i-1}
a_i	mm	DH-Parameter, translation along the axis x_i of frame F_{i-1}
$\vec{s}\ \vec{d}\ \vec{a}$	-	Unit vectors of the frame F_{FP} located at the FIS TCP
t	-	Real parameter
u, v	-	Barycentric coordinates
n_{LS}	-	Number of laser rays
π	-	Pi number
φ_{az}	°	Azimuth angle
$\varphi_{ti},$	°	Tilt angle
φ_{el}	°	Elevation angle
I_0	-	Anchor frame
I_i	-	Cloned frame
$\Delta\varphi$	°	Incrementation step of the orientation discretization scheme
N_{va}	-	Valid solution of the inverse kinematic
N_{th}	-	Possible solution of the inverse kinematic
$P_{1\ldots6}$	-	Satellite points
$S_i,$	-	Scannability index
Δd	mm	Sampling parameter of the FIS working space
$k(s)$	-	normal curvature of a natural parameterized function
$k_{e_1e_2}$	-	Curvature between two edges in a mesh
α_{Cone}	°	Aperture angle of the scannability cones
α_{min}	°	Minimal value of the scannability cone aperture angle
α_{max}	°	Maximal value of the scannability cone aperture angle

Symbol	Unit	Description
α_{opt}	°	Optimal aperture angle of the scannability cones
r_{ii}	-	Entries of a rotation matrix
$p_{x,y,z}$	mm	Cartesian position

1.1 Motivation

1 Introduction

1.1 Motivation

Nowadays, production is characterized by mass customization manufacturing that aims at providing customers with the product that satisfies their specific needs (REINHART ET AL. 2003, PILLER ET AL. 2006). In such a context, preventing flawed products from reaching the consumers is more crucial than ever (SUPERVILLE & GUPTA 2001, HEISS 2009). Shipping poor-quality products will result in customer complaints and dissatisfaction, return of goods sold, loss of image and, by association, loss of customers—inevitably leading to dwindling market share.

Since the prevention of quality troubles cannot exclusively be assured by management methods (DUTSCHKE & KEFERSTEIN 2008 p. 6), it is crucial to identify and recover defective items as early as possible (KUNZMANN ET AL. 2005, WULFSBERG & CLAUSING 2008). For this purpose, metrology equipment is deployed across the production lines to locally assess the quality of manufactured parts before they reach the next processing step (BOILLOT & UOTA 2002). In order to accommodate the diversity of characteristics on a single part or to keep abreast with the manifoldness of different parts flowing down a production line, this metrology equipment must display a high degree of flexibility. In this way, it can be swiftly adapted to changing quality inspection tasks (SOLOMAN 2010 p. 380, SCHMITT ET AL. 2011).

Since Quality Inspection (QI) may take different forms (BOSCH 1995 p. 271), it is emphasized that throughout this work the term QI refers exclusively to assessing the dimensional conformity of parts against their nominal specifications. In dimensional QI, acquiring shape data of the part under investigation is the key activity. This is accomplished by precise digitization systems that are able to capture a digital copy of physical parts at different levels of granularity. Based on the hardware used, digitization systems can be categorized into contact and non-contact techniques (WECKENMANN ET AL. 2009). The former techniques draw on accurate mechanical movement stages and touch probes to achieve precise measurements. Their main disadvantage is that they are only suitable for gauging parts that need a small number of sampled data. In contrast, non-contact techniques, to which Flexible Inspection Systems (FISs) belong, can be applied to digitize nearly all kinds of free-form surfaces (PRIETO ET AL. 2002, LANZA ET AL. 2010). Despite this advantage, the widespread deployment of FISs on manufacturing floors, especially in the context of small or medium batch production, is still far from being reached. This can be attributed in large measure to the extensive amount of time and human effort required to set up and reconfigure FISs (REINHART & TEKOUO 2009). Indeed,

reconfiguring robot-based production facilities always involves time-consuming manual programming activities to adapt the robots to the new process realities (SOLLER & HENRICH 2009). Those activities take up a large portion of manufacturing lead time and sizably increase changeover time (SCHRAFT & MEYER 2006).

1.2 Flexible Inspection Systems (FISs)

1.2.1 Overview

Most of the time, FISs perform inspection tasks in a standalone environment on flexible production lines. In Groover (2001 p. 734), FISs are defined as computer-controlled handling systems equipped with a sensor head which is capable of capturing quality relevant information. The measuring technology embodied in the sensor head is mostly optical in nature and is based on the principle of active triangulation (EICHHORN 2005 p. 30). Triangulation is the process by which the position of a point observed by two calibrated camera can be determined by considering basic right triangle relationships between the targeted point and the two cameras. In active triangulation, one of the two cameras is replaced by a light source that marks the scene with a specific illumination pattern (SANSONI ET AL. 2009). As shown in figure 1.1, the projected light pattern can be 1D (laser point), 2D (laser line), or 3D (light stripes) (ALMEIDA 2007 pp. 9–13).

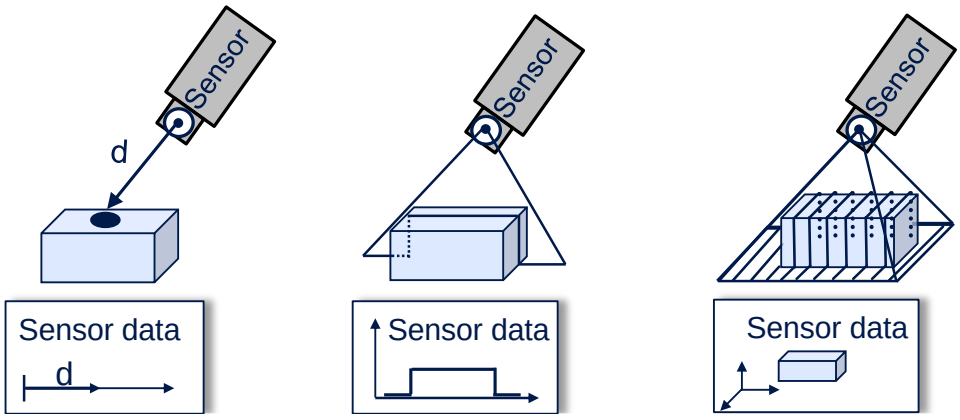


Figure 1.1: Typical light patters projected by optical sensors

Accordingly, the sensor readings are range data of the same dimensionality as the light pattern. One of the most significant advantages of optical sensors is their ability to capture precise and dense range data in a short amount of time (CHEN ET AL. 2000). For an in-depth survey of optical methods and sensors, readers are directed to BLAIS (2004), LI & GU (2004) and SANSONI ET AL. (2009).

With regard to the handling system for carrying the LSS, there is virtually no technical restriction on its topology. It can take the form of a turntable, a translation stage, a Coordinate Measuring Machine (CMM), or an Industry Robot (IR). In the early days of FISs, CMM were preferred as handling unit. Meanwhile, IR-based FISs have established themselves on the manufacturing floor, especially in large-batch production (SCHNEEFUß & PFEIFER 2004, BERTAGNOLLI 2006 p. 18, ZÄH & RASHIDY 2006). For companies already relying on IRs in manufacturing processes other than QI, opting for IRs as sensor handling units yields synergetic effects resulting from the use of a common equipment type across the plant. Examples of such synergistic effects include, a faster learning curve for the employees, a moderate increase in training costs, and lean spare part keeping (ROOKS 2001). According to EURON (2007 p. 5), IRs are de facto expected to increasingly become the "central portion of investments" in key manufacturing sectors such as the automotive Industry. To account for this growing popularity of IRs, the research carried out within the framework of this thesis focuses solely on IR-based FISs. Henceforth, the term FIS will refer exclusively to IR-based FISs, unless otherwise specified.

1.2.2 Detailed Description of a FIS

Figure 1.2 illustrates the typical setup of an FIS workcell. Its core components consist of a short-range light-section sensor, an IR, a calibration device, metrology software and the Workpiece (WP) to be measured. The functionality of these components is described in detail below.

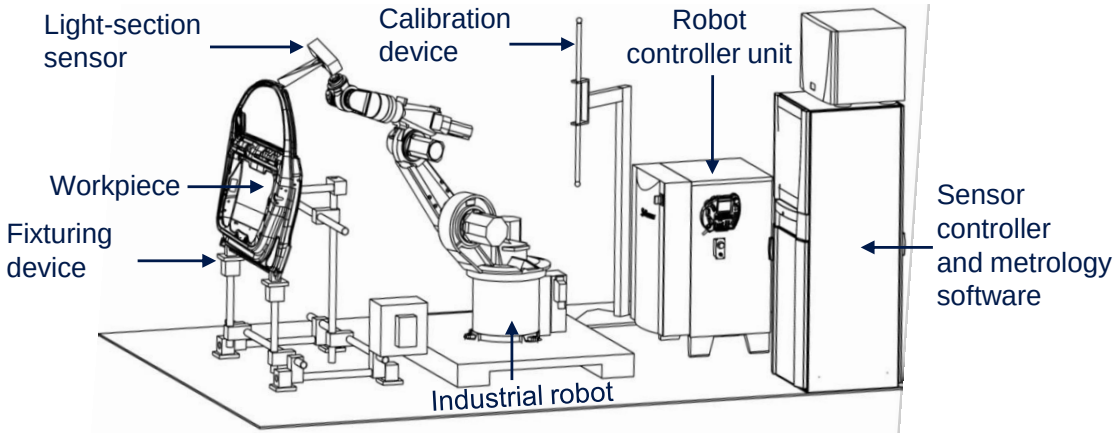


Figure 1.2: Typical FIS configuration

1.2.2.1 Light-Section Sensor

Light-Section Sensors (LSSs) use the triangulation principle to swiftly measure variations in distance between themselves and a WP's surface (SCHWENKE ET AL.

2002). Figure 1.3 schematically depicts their functional principle, which can be summarized as follows. First, a plane of red light rays, produced by a low-power laser source, is focused on the surface of the WP to be measured. The intersection of the laser plane and the WP's surface is a 2D distorted laser stripe also called *light section* or *scan profile*. This 2D scan profile is then captured by the camera lens and focused on a position-sensitive detector located in the sensor casing. There, the captured scan profile image is post-processed to obtain its digital replica.

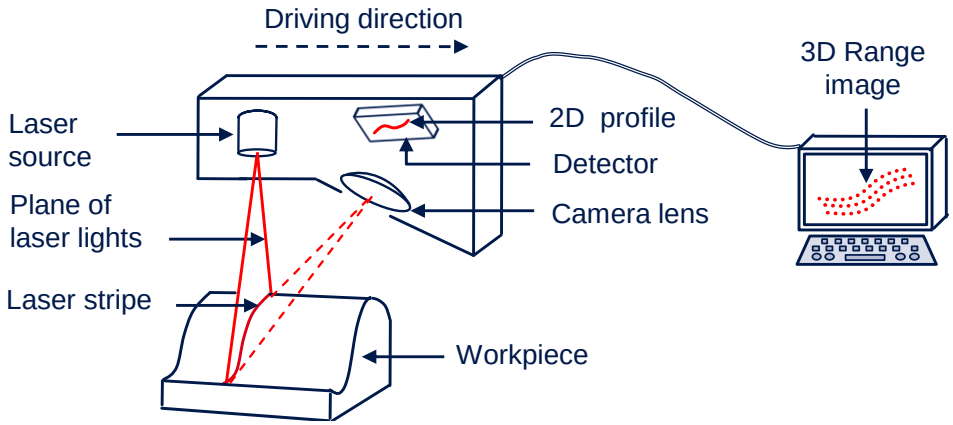


Figure 1.3: Function principle of LSSs

Since a single sheet of light is insufficient to gather the range information for a complete WP, a handling device is required to mechanically drive the sensor in overlapping swathes across the surface being digitalized (GELFAND 2006 p. 4). The missing third spatial dimension of each scan profile is obtained from the handling device encoders. Compiling the three data dimensions yields a 3D range image[1] that approximates the WP surface topology.

In terms of existing systems implemented in the industry, LSSs represent the most popular approaches for optical measurement systems (NURRE 2000 p. 7, HUNG 2005 p. 11). In addition, they are considerably smaller and lighter than their 3D counterparts, e.g. photogrammetric, projection fringe sensors. Thus, they are better suited to access filigree details or serpentine artifacts on WPs that may be not visible from far away. The measurement accuracy of LSSs is claimed to typically vary from 20 µm to 50 µm (BLAIS 2004). When set against the required manufacturing tolerances of sheet metal parts, which typically range from 0.5 mm to 1.5 mm (KUNZMANN ET AL. 2005), it is obvious that LSSs fulfill the "golden rule of metrology." The latter dictates that uncertainty of metrology equipment must not exceed one or at least one-half order of magnitude of the characteristic's tolerance under consideration (WECKENMANN ET AL. 2001, PFEIFER 2002 p. 84).

[1] A range image refers to an array of 3D points generated by an optical sensor after a measurement session. In some of the literature, range images may also be referred to as "point clouds."

1.2.2.2 Industrial Robots

The powerful potential of LSSs cannot be achieved until they are integrated into a robot-based handling system. The latter substantially increases their work volume and gives them greater flexibility (NURRE 2000). Since at least six Degrees of Freedom (DoF) are required for positioning an LSS at an arbitrary location, FISs are mostly based on six-axis IRs. With regard to the utilization of those IRs in metrology applications, three approaches can be distinguished:

- The IR may act as a handling device for the WP, which implies that the sensor is fastened to a fixed-space frame or, in rare cases, to another robot (ABBATE 2005). From a metrological point of view, only good repeatability of the IRs is needed here.
- The IR can also serve as a passive carrier for the sensor. In such a configuration, the IR's kinematic chain does not have any influence on the measurement results and an external tracker is required to monitor the sensor pose during the measurement (HEDENBORN & BOLMSJÖ 1995). This approach is limited by the fact that a free line of sight has to be guaranteed between the sensor and the monitoring device. This can be realized, if at all, at the expense of system complexity.
- Last but not least, IRs can act as a measurement instrument as well. This category, into which the FIS investigated in this research also falls, is the most exigent with respect to IR accuracy—the reason for this being that the IR accuracy sets the limits for overall system precision.

In conclusion, it should be stressed that IRs present both interesting opportunities and tough challenges for metrology applications. One of these challenges involves the positioning error of IRs that adversely impacts the overall accuracy of FISs (ROOS & BEHRENS 1997). In order for IRs to be fit for use in metrology, their inherent positioning inaccuracies have to be reversed, at least up to a certain extent.

1.2.2.3 Calibration Equipment

Although robot-manufacturing companies are not eager to publish any statistics on the absolute accuracy of their products, several publication, e.g. REINHART ET AL. (1998), BEYER (2005 p. 7), and ABDERRAHIM ET AL. (2007) reveal that the scale of the error between the actual and the desired position of off-the-shelf IRs can be as high as one to several millimeters. That is two orders of magnitude higher than the accuracy of LSSs. Logically, one may expect this to technically bar IRs from being deployed in metrology chains. Fortunately, the accuracy performance of IRs can be enhanced using robot calibration techniques (VERL ET AL. 2008).

Robot calibration refers to procedures by which the deviation of some key parameters of a kinematic structure relative to default settings—e.g. link dimensions, compliances, and relative arrangements, the joint elasticity—can be offset by non-mechanical means (GRÄSER 1999 pp. 19–23). These deviations may be induced by manufacturing tolerances, thermal effects, mechanical wearing, inaccurate internal models used in the controller unit, to name but a few (VERL 2009).

The calibration process consists of the modeling of the robot structure, the identification of its non-redundant parameters and finally, the comparison of the identified parameters with the default parameters (BONGARDT 2003 pp. 39–49). The identification process is driven by a set of redundant measurements performed by means of an external non-contact measurement device. The outcome of the calibration process is an individual signature of the IR. This signature is used to adjust motion commands of the IR in such a way that it comes as close as possible to an intended location. According to BEYER & WULFSBERG (2004), calibration techniques can help raise IRs positioning accuracy up to average errors below 0.1 mm.

Figure 1.4 shows the setup of the technique used to calibrate the IR which will be later on integrated in the demonstration platform used for testing purposes in this research (see chapter 8). This calibration setup features a high accuracy laser tracker that measures the position of a retro-reflector target fixed at the IR's tip. For an in-depth explanation on calibration techniques, readers are directed to MOORING ET AL. (1991) or BERNHARDT & ALBRIGHT (1993).

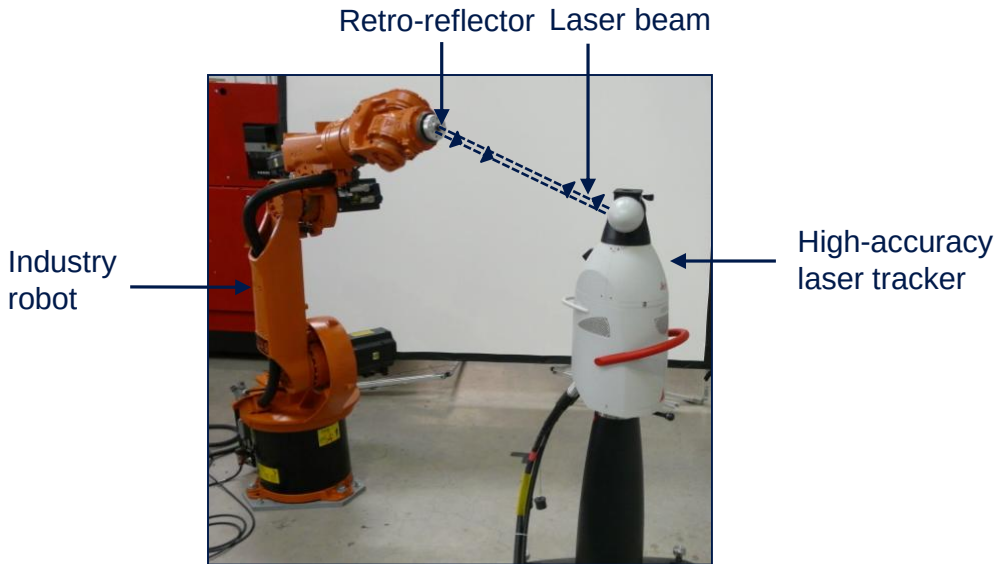


Figure 1.4: Setup for calibrating an IR

1.2.2.4 Metrology Software

The 3D range image obtained after the digitization process is transferred to metrolo-

gy software where the conformity of the part with regard to a number of predefined tolerance margins is assessed. Before the conformity grading can take place, the 3D range images are subjected to some pre-processing steps to reduce the amount of data, remove outliers and merge multiple data patches into a common coordinate system (NIKON 2008). The quality assessment itself is performed by juxtaposing and comparing the 3D image with the nominal WP's CAD model. The data comparison step yields a customized color-shaded image (OTTOVISION 2009), which reports surface deviations relative to the CAD model (see figure 1.5). Information about critical features such as edges, holes, slots and studs can be generated as well.

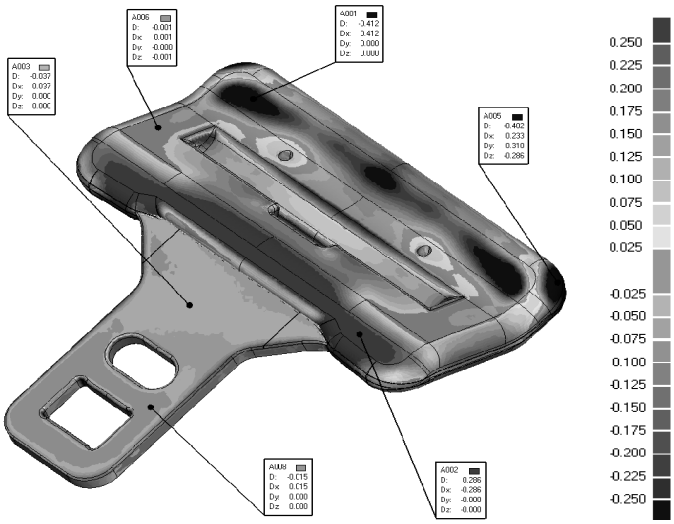


Figure 1.5: Color-shaded QI report (OTTOVISION 2009)

1.2.2.5 Main Application Fields

FISs are primarily deployed for the digitization of sheet metal parts for QI or process control. Digitization activities are carried out during several phases of the product development cycle, which involves development, prototyping, as well as pilot and production phases. FISs have also already found their way into niche applications. For example, they are deployed in the wood-logging industry to measure logs and help prescribe wood cutting profiles for optimum yields. Selective part assembly also represents an interesting application field for FIS. Selective assembly is the means by which components are assembled from subcomponents, such that the final assembly satisfies higher tolerance specifications than those used to make its subcomponents (PUGH 1986). The final product is assembled from subcomponents of mating pairs, which were measured to meet the required specifications as closely as possible. This technique is common practice in hi-tech manufacturing as found in the bearing industry. More detailed information about the use of FISs is available in (EICHHORN 2005) and (GALANULIS 2005).

1.3 Delimitation of the Problem Domain

Although in this research FIS was approached with generality in mind, this work is tailored to FISs devoted to sheet metal QI. Sheet metal is a ubiquitous material with wide applications in myriad industrial branches ranging from electronics, medical, and aerospace up to automotive and consumer goods. Currently, over half of the total metal production is used for goods made of sheet metal (STREPPEL ET AL. 2008). This popularity is justified by the fact that highly sophisticated sheet metal parts with various levels of complexity can be produced at acceptable costs (WECKENMANN & GABBIA 2006, RAO 2007).

It is also worth emphasizing here that, in sheet metal manufacturing, keeping defective parts from reaching the next processing step may be more crucial than in other manufacturing branches (ROOKS 2001). A reason for this is that products made from sheet metal generally comprise many components combined into a single entity that cannot be easily disassembled. Hence, any defective components can render the whole assembly useless (REINHART & RASHIDY 2008). Failing to detect faulty items far beyond the root cause could create costly difficulties such as unnecessary stoppages of machinery and costly rework on the production floor (MÖNNING 2006 p. 3). To put it in economic terms: the financial costs for the elimination of defects increase by about a factor of ten for every further phase of the product life-cycle traversed by the faulty item (KUNZMANN ET AL. 2005). Figure 1.6 schematically expresses this notion, which is also commonly referred to as the "rule-of-ten."

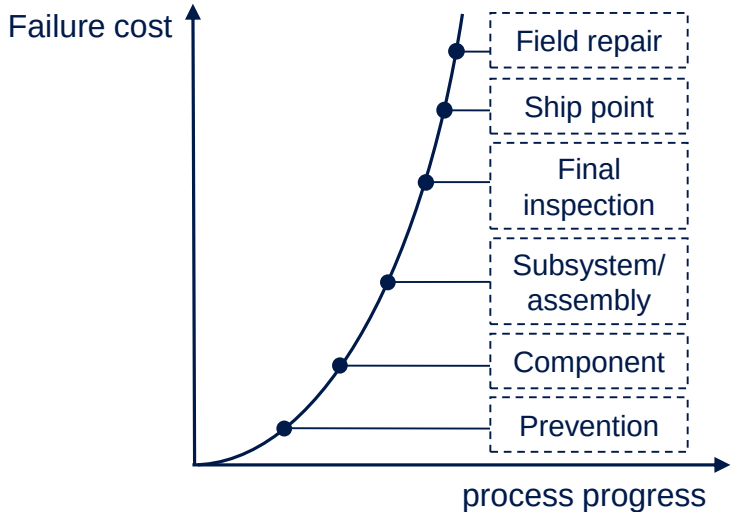


Figure 1.6: Failure cost as a function of detection point (derived from CAMPANELLA 1999)

1.4 Problem Statement

As indicated in section 1.1, programming FISs is cumbersome and time-consuming especially when it comes to batch production with low-volume and large number of part variants. The causes for this programming complexity lie in the multitude of restrictive conditions imposed by both the LSS and the robot manipulator in order to perform a digitization task (REINHART & TEKOUO 2009). As for the LSS, keeping Areas of Interest (AoI) on the WP surface within the narrow and invisible sensor measuring window is mandatory and not a trivial task (SCOTT 2009). In addition, unhampered access to the WP for the sensor housing should be provided during the digitizing activities. Failing to do so may either expose the LSS to collision with the WP or cause a temporary occlusion of the laser light path or camera view field. From the perspective of the IR, dynamic constraints in terms of velocity, acceleration and torque limits have to be satisfied (CONSTANTINESCU & CROFT 2000). Equally relevant for the IR are kinematic restrictions such as reachability, collision and singularities (HESSELBACH ET AL. 2005).

Currently, FIS programming is usually performed by metrologists or other highly skilled specialists who draw on their personal experience—as well as trial-and-error approaches—to manually generate adequate digitization programs (WULFSBERG & CLAUSING 2008). Relying on such tedious and time-consuming programming methods often results in over- and under-scanning the AoI, losing some interesting filigree regions on the range image as well as producing suboptimal or inaccurate movements of the FIS (ELMARAGHY & YANG 2003, REINHART & TEKOUO 2010). The disadvantageous impact of manually programming FISs could not be clearer: commissioning costs which exceed that of the FISs, longer changeover times and suboptimal process cycle times. These drawbacks may not harm large-scale manufacturing corporations since the reconfiguration cost are spread over the mass production and a relatively long production time. But for small batch manufacturing companies, this modus operandi is economically detrimental and, over the long haul, unsustainable. Therefore, there is still a strong incentive to ease the reconfiguration burden of FISs in order to give them access to wider or new market opportunities (NAUMANN ET AL. 2009). As claimed by SON ET AL. (2003), an appropriate approach to tackling robotic programming issues is to develop smart software tools that can automatically generate optimized robot programs with minimal interaction of non-trained human operators.

1.5 Contribution and Significance of this Dissertation

Based on the above problem statement, the overall objective of this research is to

develop a methodology for automatic programming of FISs, with emphasis on those deployed in low-volume, high-variant manufacturing environments. In order to put the results of the research into a true industrial perspective, the methodology presented will not be restricted only to theory, but subsequently implemented as a prototypical automatic programming tool.

The application of the proposed methodology is believed to raise the flexibility and attractiveness of FISs to another notch. In fact, instead of having to cope with various levels of complexes functionalities and having to familiarize themselves with sophisticated equipment, operators facing an inspection task will be led through it by the programming system and so be able to easily (re)program FISs in a short time period. The most salient contributions of this research, partially published in international scientific journals—see REINHART & TEKOUO (2009), REINHART ET AL. (2011)—can be summarized as:

- the conception and development of a method to support an advantageous positioning of WPs within the robot workcell;
- the development of a CAD-based task description method;
- the conception and development of an automatic sensor trajectory planner that accounts for the FIS's limitations.

With regard to the broader significance of this work, although motivated by a premise prevailing in batch manufacturing, it is also pertinent for FISs deployed in mass production. As stated by LOZANO-PEREZ & TAYLOR (1989), the availability of Automatic Programming Systems (APSs) will inevitably help clear programming hurdles in large-scale production. Moreover, the methodology proposed is also expected to be conceptually applicable to other trajectory-oriented tasks such as arc-welding, high-pressure water cutting, bonding and sealing—to name just a few.

1.6 Structure of this Dissertation

The preceding sections have given a glimpse into the motivation of this doctoral dissertation. It structure can be summarized as follows:

Drawing on the insights provided in *chapter 1*, the necessary background on the core topics of this research is given in *chapter 2*. Additionally, and insight on existing work in the area of sensor planning and robot programming will also be presented there. *Chapter 3* serves as an anchor for the remaining chapters and uses the findings of the literature review as a basis for designing the new programming methodology.

The development of the system components and the relevant techniques used are

subsequently described in *chapters 4, 5, 6* and *7*. *Chapter 4* is dedicated to the development of a simulation module that serves as a knowledge basis for the algorithms and methods to be subsequently applied. Several technical terms and definitions that will be used in later chapters are also introduced there. The first part of *chapter 5* is devoted to the introduction of a new performance metric that characterizes the FIS's capabilities within its workcell. Based on this metric, a method by means of which the WP under investigation can be appropriately positioned within the FIS's workcell will be elaborated. *Chapter 5* also includes a discussion of a method to easily describe an inspection task based on the CAD model of a WP, will be discussed.

Chapter 6 concentrates on the automatic derivation of scan paths as well as scan trajectories from the task description. *Chapter 7* is concerned with the planning of constrained robot movement to realize the scan trajectory. As the capstone of this work, *chapter 8* contains details about the implementation of programming methodology into an APS. Additionally, the performance of this programming system is demonstrated using an industrial example from the automotive industry. *Chapter 8* concludes by shedding some light on the technical and economical rationale of the presented methodology. A summary of the core content as well as the prospects for future research directions are given in *Chapter 9*. Figure 1.7 graphically illustrates the overview of this work elucidated in the previous lines.

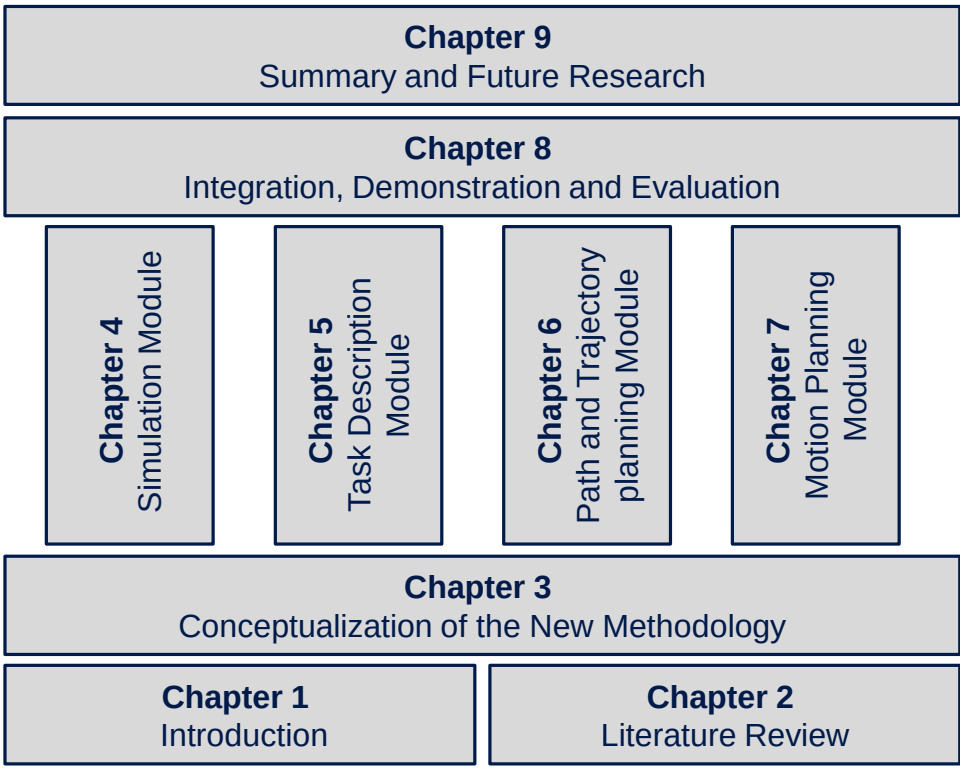


Figure 1.7: Structure of this research

Before closing this introductory chapter, it should be mentioned that, except for those discussed in the literature review chapter, fundamentals essential for the understanding of methods and algorithm developed in this research will be explainned directly as needed. This should prevent the reader from having to search for explanation across the manuscript. It should also be noted that both male and female genders are intended when only the masculine form of nouns or pronouns is used.

2 Literature Review

2.1 Overview

This chapter outlines literature relating to the key topics of this research and is organized in three main sections. In the first section, existing attempts to overcome the challenge of automating sensor planning are surveyed. The second section deals with paradigms related to the programming of IR systems, while the third and last one summarizes the findings surveyed in the previous sections and highlights the motivation behind this research. Before elaborating on these topics, it should be noted that sensor planning and robot motion programming are themselves separate fields of study with a correspondingly large body of work. Moreover, solutions for sensor planning vary greatly according to the sensor type and the nature of the task at hand (REED 1998 p. 64). Consequently, the volume of existing literature referred to in this review was narrowed down with respect to its relevance for this research.

2.2 Sensor Planning

2.2.1 Background

2.2.1.1 The Challenging Task of Planning Sensor Viewpoints

Due to the function principle of LSSs, their measuring windows are relatively small compared to the size of WP they are intended to digitize. Hence, in order to collect enough data, a large number of viewpoints, i.e. discrete sensor positions and orientations in space, from which the WP is observed, are necessary. This calls for sensor planning strategies that are able to select and direct the sensor to viewpoints from which it can see appropriate sections of the WP under investigation. In the seminal works of TARABANIS ET AL. (1995A), the field of sensor planning is comprehensively surveyed. There, sensor planning is defined as "*developing automatic strategies to determine the geometric (e.g. position, orientation) and optical (e.g. focal length, resolution) sensor parameters, in order to carry out a measurement task.*" In the same source it is argued that effective sensor planning strategies must consider the physical characteristics of the sensor. Indeed, sensors are subjected to physical limitations that may or may not make certain AoI on the WP's surface observable from a specific vantage point. For this reason, TARABANIS ET AL. (1995A) inferred that quantifying the relationship between a WP and an observing sensor necessitate model-based approaches. Figure 2.1 displays the general structure of a sensor

planning system as conceptualized in (TARABANIS ET AL. 1995A).

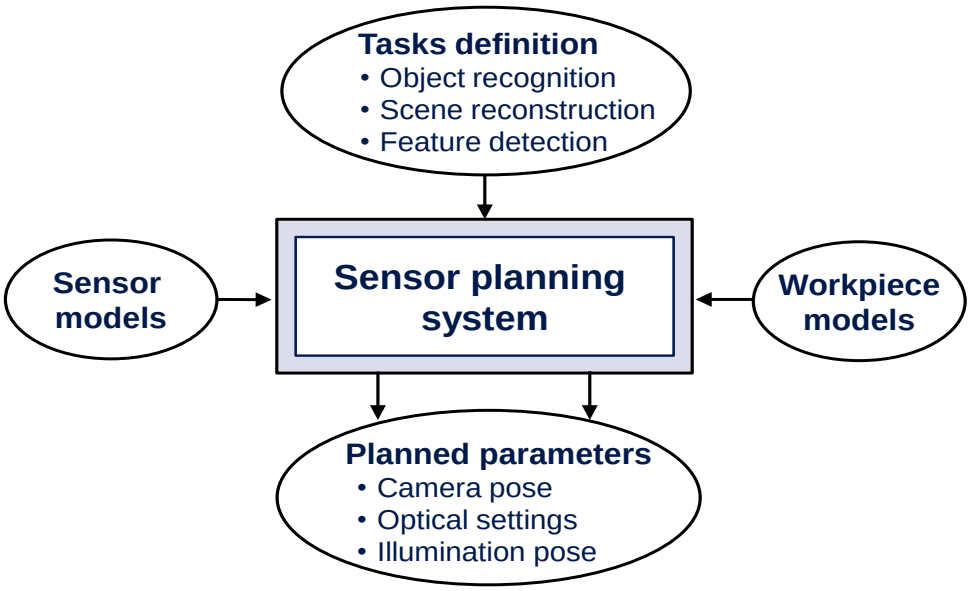


Figure 2.1: Sensor planning for computer vision

2.2.1.2 Sensor Planning in FIS

In FISs currently deployed, scan trajectory are still determined by trial-and-error planning techniques (WONG & KAMEL 2004, DERIGENT ET AL. 2007). Sensor locations and settings are chosen by experienced metrologists who rely exclusively on their own experience and knowledge. However, manually generating adequate and efficient scan trajectories for complexly shaped WPs is not trivial, notwithstanding how experienced an operator may be (SCOTT ET AL. 2003). While challenges still remain, robust sensor planning algorithms have already been developed for CMM-based FISs. This is not surprising since sensor planning for CMM-based FISs bears similarities to the automatic generation of programs for NC machining, which has been a longstanding research topic for decades. Unfortunately, sensor planning approached tailored for CMM-based FISs are not transferable offhand to robot-based FISs. One reason for this is that CMMs' trivial kinematics do not need to be explicitly accounted for at the time of sensor planning (BERNARD & VÉRON 1999, ELMARAGHY & YANG 2003).

2.2.1.3 Basic Imaging Constraints: Visibility

Regardless of the task or the subsequent processing stages that await the images acquired by an optical sensor, there is a basic requirement that a sensor viewpoint must satisfy for the resulting image to be useful: the *visibility constraint*. In the context of LSSs, this constraint requires an unobstructed line-of-sight into the

targeted AoI on a WP surface for both the laser source and the receiving camera optic of the sensor (PITO 1999). This constraint decisively impacts on the success or failure of a digitization task, especially in the case of complex shaped WPs. Indeed, since a digitization task on complex shaped or large WPs cannot be carried out with a single sensor shot, the sensor has to be moved around the object, from one viewpoint to the next. The vantage points to be visited by the sensor result from searching for candidate solutions in an area of the free space surrounding the WP called *visibility space*. The latter labels those regions of the free space from which the sensor can see specific AoI on the surface of a given WP. In other words, the *visibility space* delimitates a portion of the free space from which each sensor viewpoint fulfills the *visibility constraint.* The complement of the *visibility space* is called *occluded space* and consists of the region of the free space from which any AoI on a WP may be either partially or fully obstructed. It goes without saying that the visibility space or the occluded space may be empty.

2.2.1.4 Taxonomy of Sensor Planning Methods

Based on the research by MALAMAS ET AL. (2003), SCOTT ET AL. (2003) and LI & GU (2004), existing sensor planning approaches can be characterized as online or offline depending on the amount of *a priori knowledge* available about the digitizing environment.

Online approaches

Sensor planning techniques that fall into this category rely on the premise that nearly everything about the WP to be digitized is unknown beforehand, except its bounding volume. These planning methods operate by first gradually building a coarse model of the WP under investigation through a systematic exploration of the bounding box that roughly delimits the WP's volumetric shape. To do so, surface following techniques (LARSSON & KJELLANDER 2008) or volume partitioning procedures (REED & ALLEN 2000, CALLIERI ET AL. 2004) come into play. The "Knowledge" derived from this preliminary modeling phase is then reused to design a subsequent planning phase where a more accurate shape of the WP is collected.

Online approaches to solving the sensor planning problem correspond more to active vision than to classical sensor planning (PITO 1999). The overall objective here is to have a complete range image of the WP. Such complete digital images are intended more to be used for reverse engineering purposes rather than for QI applications. According to MARSHALL & MARTIN (1993 p. 313), online methods are unsatisfactory because they generate a lot of unsampled areas on the WP surface that have to be subsequently completed by manually sweeping the sensor over them. Another major disadvantage of online approaches is that some viewpoints

may be redundant, engendering overlapping sensor sweeps. Moreover, it is also reported in the same literature that QI requirements impose conditions on how the viewpoints are chosen, i.e. arbitrary viewpoints cannot be trusted. From this it is obvious that exploiting detailed knowledge about the digitizing environment will improve the performance of sensor planning algorithms (GERMANI ET AL. 2009).

Offline approaches

Offline sensor planning approaches, into which category the majority of the sensor planning research falls, operate based on a model of the digitizing environment. Such models are usually available as 3D CAD, which is why offline sensor planning is also referred to as CAD-based sensor planning. A major advantage of using CAD models for sensor planning purposes lies in the parallelizability of sensor planning activities to product design. The generic procedure adopted by offline sensor planners can be broken down into three sequential activities, namely:

- the selection of suitable viewpoints allowing robust coverage of the AoI;
- the sequencing of the viewpoint visiting order to form a scan path while taking into account efficiency considerations;
- the generation of sensor scan trajectories suitable to the sensor handling system characteristics.

Since the programming methodologies to be developed in this work can be categorized as offline, the last three items merit more elaboration. In order to do so, existing approaches addressing these activities will be examined in greater detail in the following sections.

2.2.2 Viewpoint Selection

From a high-level perspective, existing work in the area of candidate viewpoint selection can be characterized by the number of stages the viewpoint synthetization algorithms are composed of. For the purpose of this research, it is sufficient to differentiate between *monolithic* and *multi-staged* viewpoint selection schemes.

2.2.2.1 Monolithic Viewpoints Selection Schemes

Monolithic viewpoint selection schemes can roughly be categorized in *synthesis* and *theoretical approaches*.

Synthesis approaches

Synthesis planning approaches address sensor view planning analytically by formulating the task as a constraint satisfaction problem. The digitization constraints are modeled by means of nonlinear polynomial functions a various mixed systems of

equality and inequality bounds. This brings powerful knowledge to bear that can be utilized to answer the sensor planning question. Furthermore, mathematically formulating the task implicates that, well-proven algorithms for solving constrained nonlinear optimization problems can easily fitted in. The earliest investigation in this area can be traced back to the work of COWAN & KOVESI (1991). They presented a conservative viewpoint selection technique that consists of individually synthesizing geometric relationships between the sensor's poses and a restricted set of digitization constraints. These constraints are subsequently combined to create a unique objective function. This work was later further extended by TARABANIS ET AL. (1995B). They considered additional parameters in the formulation of the viewpoint selection strategy and relaxed some DoF on the sensor location.

In general, it can be stated that synthesis planning schemes necessitate an understanding of the causal relationships between the parameters to be planned and the goals pursued. In this way, a deeper insight is provided into the sensor planning problem. Moreover, such planning schemes also exhibit the advantage of simultaneously optimizing competing factors over a multidimensional visibility space. However, in some cases, analytical relationships between sensing parameters and planning goal simply do not exist (DUNN & OLAGUE 2004), rendering this approach impracticable. In addition, their applicability also suffers from nonlinearity, computational difficulties and convergence issues, which often arise (SCOTT ET AL. 2003).

Theoretical approaches

Theoretical approaches tackle the viewpoint selection problem by drawing on some well-studied scientific challenges. Proceeding that way opens the door to a myriad of established scientific methods to solve sensor planning problems. The idea of treating the viewpoint selection problem from a theoretical point of view goes back to TARBOX & GOTTSCHLICH (1995). In their seminal work, which has been improved upon by SCOTT (2009), they hypothesized that viewpoint selection is isomorphic to the set covering problem. The rationale behind their idea was to capitalize on the rich algorithms repertoire available for discrete combinatorial optimizations. In contrast, various researchers such as URRUTIA (2000), DANNER & KAVRAKI (2000) and GONZALEZ-BANOS & LATOMBE (2001) have viewed sensor planning as a robotic instance of the art gallery problem. The latter originates from a real-world problem of guarding an art gallery with the minimum number of guards capable of observing the whole gallery's painting (O'ROURKE 1987). However, the art gallery problem is not easy to solve. Even its 2D version, i.e. inspecting all the interior edges of a polygon with minimum vantage point belongs to the class of NP hard[2] problems (CULBERSON & RECKHOW 1994).

[2] Refers to a type of problems in computational complexity theory that cannot be solved in polynomial time.

The art gallery approach has actually been discarded as a solution for sensor planning in 3D space (SCOTT 2009). Many researchers such as LI & LIU (2005) and TORABI ET AL. (2007) have applied an information theoretical method for viewpoint planning. To encode the visibility of viewpoints, they introduced a theoretical measurement termed viewpoint entropy. This measurement is based on information theory and can be interpreted as the amount of information seen from one single point. They defined the best viewpoint as the one that has maximum entropy. For simple polyhedral objects, theoretical approaches may offer intriguing theoretical possibilities. However, they fail short when it comes to WPs with complex shapes.

2.2.2.2 Multi-Staged Viewpoints Selection Schemes

Even though most viewpoint selection methods can be characterized as monolithic, utilizing a single technique throughout the process is unrewarding (SCOTT ET AL. 2003). According to HALL-HOLT (1998) it is more advantageous to divide the planning problem into hierarchical stages and consider different techniques for each of them. In doing so, restricting and computationally intensive techniques can be limited to subsets of the overall problem. Depending on the domain of reasoning about the discretization cue of the view space, multi-staged viewpoint selection schemes can be grouped in the categories listed below.

Slicing plane approach

SHU & XI (1999) developed an algorithm to slice the Boundary Representation (B-reps) of a WP CAD model into adjacent sections, as illustrated in figure 2.2.

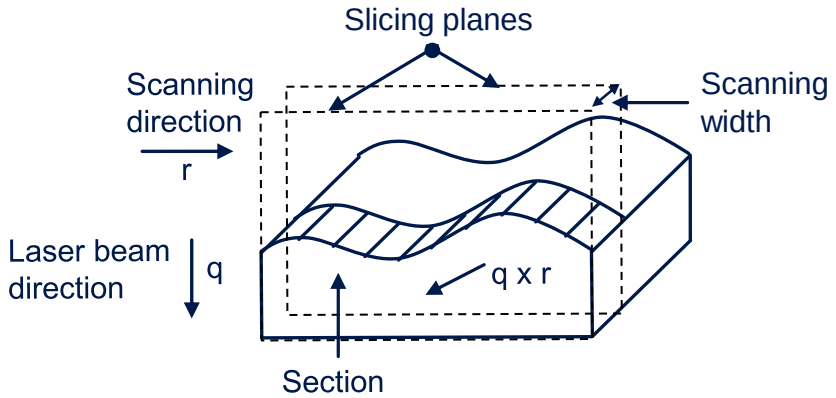


Figure 2.2: Slicing plane approach. r denotes the scan direction, q the laser beam direction, the surface to be scanned is sliced by a set of cutting planes whose normal is parallel to $q \times r$.

The slicing planes are spotted in such a way that each section can be covered by the laser stripe width within a scan pass. Subsequently, their viewpoint selection strategy involves acquiring long strips of profile along the sliced section by keeping

the sensor in close proximity to the surface at all times and simultaneously satisfying visibility constraints. However, the relevancy of this work is restricted by the limitation of the sensor movements to translation only. Unfortunately, translational sensor movements by themselves are insufficient to cover free-form shaped WPs. In addition, having only three translational DoF makes the placement problem much simpler than in the case where rotational DoF are considered.

Part parametric surface decomposition approach

PRIETO and his co-researchers investigated a method to find viewpoints needed to completely digitize the WP while also optimizing the accuracy of the measurement data (PRIETO ET AL. 2003). For this purpose, the CAD model is decomposed into NURBS[3] parametric surfaces, and projected viewpoints on each of these surfaces are generated. From the projected viewpoint, a ray is traced to find a corresponding viewpoint that is not obscured by the rest of the CAD model. Since the sensor used is a single-point range finder, i.e. the sensor only provides one distance reading to each point on the surface, the range images collected are quite sparse unless they are densified a posteriori at the expense of the process time (DERIGENT ET AL. 2007).

Local and global admissible direction approach

The viewpoint planning approach taken in (LEE & PARK 2000) sparsely samples the parametric surfaces of the CAD model. This pre-processing step serves to reduce computational time. For each sampled point, a *local admissible direction* (LAD) is calculated, with regard to several scan constraints (see figure 2.3). Resulting scan directions, also known as *globally accessible directions* (GADs), are then obtained by performing Boolean operations on the pre-computed LADs.

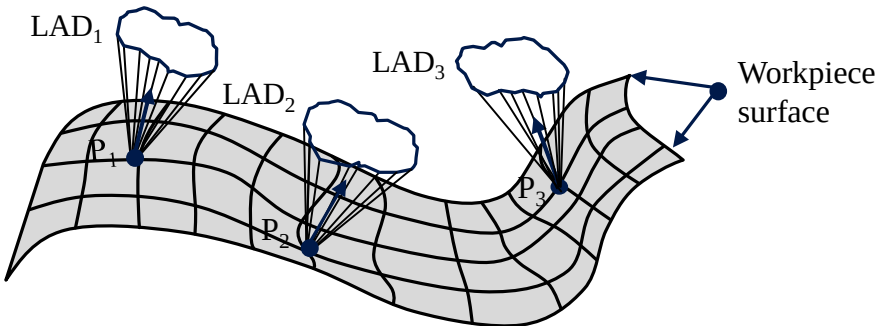


Figure 2.3: Local admissible direction (LAD)

However, this method is complex and computationally expensive. The measuring system used relies on a three-axis rotary table in order to set up the part according to each scan direction. In order to reduce the number of sensing directions, SON ET AL.

[3] NURBS stands for Non-Uniform Rational Basis Splines.

(2003) added the concept of critical points to the previous work, i.e. points that cannot be scanned together. In DERIGENT ET AL. (2007), a similar viewpoint selection procedure is reported for CMM-based FISs.

Part surface tessellation approach

ELMARAGHY & YANG (2003) hypothesize that partitioning meshed surfaces into patches[4] such that each of them has attributes that fulfill sensing requirements is always possible. Consequently, they propose a two-staged viewpoint selection algorithm to provide evidence for their assumption. The first stage clusters the model compound's surface into smooth patch regions based on the normal vector and the triangle's size as well as an adjustable flatness threshold (see figure 2.4).

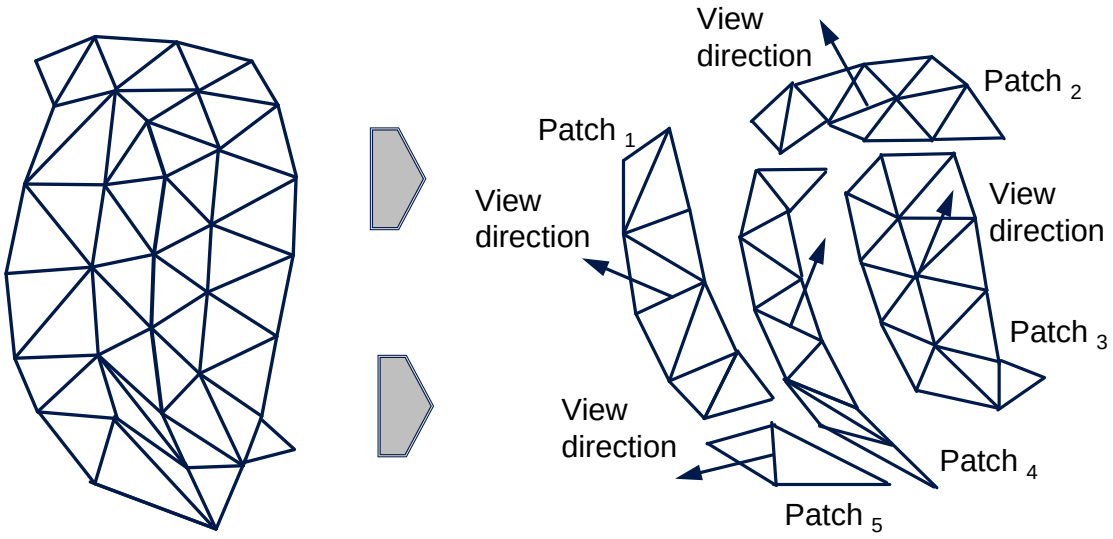


Figure 2.4: Clustering of a tesselated surface in flat patches

In the second stage, their algorithms generate viewpoints patch-by-patch by taking visibility and occlusion constraints into consideration. The viewpoint direction of the patches is obtained by averaging all the facets' normal vectors. Further studies which have used similar but slightly different procedures include SON & LEE (2003) AND FERNÁNDEZ ET AL. (2006). FERNÁNDEZ ET AL. applied back-face culling algorithms and space partitioning techniques to remove triangles occluding each other. Thus, a subset is extracted from the initial mesh that does not include those triangles whose visibility with regard to a specific viewing direction is completely blocked by other triangles.

In order to save computation time, the algorithm developed by SON & LEE (2003) does not directly act on the patches. Instead, the surface patches are finely pre-

[4] Cluster of neighboring facets having some similarities, mostly common vertices and similar normal vector direction.

sampled by points. Each point inherits the normal vector of the triangle on which it lies. Points having similar normal vectors directions are then grouped in point clusters. Subsequently, the view direction of the sensor at each point cluster is generated by calculating the global mean of the normal vectors associated to each point within the cluster. Figure 2.5 schematically illustrates a model consisting of four uniformly sampled surface patches. N_1, N_2, N_3 and N_4 denote the sensor's view direction at each patch.

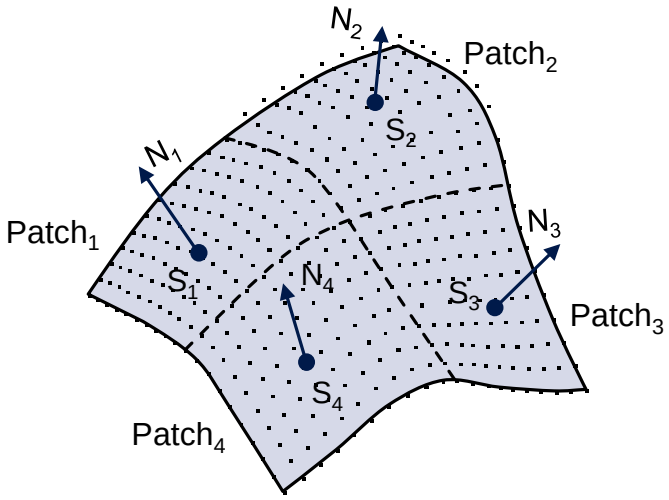


Figure 2.5: Surface patches approximated by point clusters and the resulting l scan direction

Object spatial occupancy approach:

To render computation of the scan constraints much easier, MARTINS ET AL. (2005) attempted to fit the WP CAD model into simple enclosing volumes or regular spatial subdivisions called voxel map. This is a 3D grid structure consisting of volume elements, each of which is an identical cube that essentially stores information about the space it represents by means of four attributes: unmarked, occupied, empty, and occluded. A volumetric model implemented through a 3D voxel map is generated from the object CAD model and is used to define a sensing plan composed of a set of viewpoints and the respective scan trajectories. A surface-following scheme is used to retain collision-free, efficient viewpoints.

To conclude this section, it can be stated that the multi-staged viewpoint selection methods presented above are devised only for the complete digitization of the WP with simple geometry. Additionally, it can be observed that they are specially tailored either for three-axis traverse mechanisms or rotary stage platforms.

2.2.3 Computation of Scan Paths

2.2.3.1 Combinatorial Aspect of Computing a Scan Path

Finding sensing sequences that take the sensor through a set of pre-selected viewpoints is the quintessence of the sensor path planning phase. In order to be effective, a scan path should not only enforce solution quality but also efficiency by ensuring maximum sampling coverage within an acceptable time frame (SCOTT ET AL. 2003). Once adequate viewpoints have been isolated, the time spent traversing them is the key factor that could affects the efficiency of the digitizing process. Obviously, the time required to conduct a scan plan consists of the time spent travelling to all the viewpoints, as well as the time needed by the sensor to fire the laser beam. As LSS particularly can fire their laser while being in motion, the laser firing time can be logically assumed not to constitute a significant portion of the total cycle time. Hence, optimizing the scan process time is in fact tantamount to minimizing the travelling time between the viewpoints.

The travelling time for a given path essentially depends on the visiting order of the viewpoints. For a given pre-selected set of viewpoints, there are several different sequencing alternatives that may yield similar digitization results. Brute force search algorithms, which operate by performing complete enumeration (GILMORE & GOMORY 1964), are inadequate here because of the associated high computational burden (DUNN & OLAGUE 2004). Put in mathematical terms, the running time for an algorithm implementing the brute force scheme lies within a polynomial factor of $O(n!)$. Such an approach would become impractical even for only 20 viewpoints (correspond to $20! \cong 2.4329 \times 10^{18}$ alternatives). Checking all possible tours with 50 visiting points ($50! \cong 3.0414 \times 10^{64}$) is well out beyond the combined computing power of all the machines currently in use worldwide. Fortunately, more efficient algorithms that allow the determination of the minimum time to visit n points have already been the focus of much scientific research.

2.2.3.2 The Traveling Salesman Problem

In fact, the at first glance modest-sounding exercise mentioned above happens to be one of the most investigated topics in computational mathematics: the Traveling Salesman Problem (TSP) (APPLEGATE ET AL. 2006). The TSP is also known to be NP-hard, which basically means that there is no guarantee of finding the best solution to solve it in a reasonable amount of time. Based on the exactness of the solution, algorithms to solve the TSP can be categorized as either exact or approximate (BAUM 2004 p. 21). Exact algorithms are aimed at finding the optimal solution

to the TSP. However, these algorithms are slow, especially for large-size problems. Prominent exact algorithms are *cutting planes* (EDMONDS 1965), *branch-and-bound* (DANTZIG ET AL. 1959) and *dynamic programming* (BELLMAN 1962), among others. In 1998, a research team around Applegate solved to optimality an instance of the TSP of 13,509 cities corresponding to all U.S. cities with populations of more than 500 people (see figure 2.6). The complete calculation ran on a network of 48 workstations. Scaled to a 500 MHz processor, this corresponds to 10 years of CPU time (APPLEGATE ET AL. 1998), which is an exorbitant figure. Meanwhile a solution for the exact computation of a tour with 85,900 cities—current world record—was developed at the same laboratory (APPLEGATE ET AL. 2006).

Figure 2.6: Optimal route connecting 13,509 U.S. cities with populations of more than 500 people (GATECH 2005)

Approximation algorithms, in contrast, aim at finding a near-optimal solution by balancing the trade-off between computational complexity and efficiency. They draw on powerful heuristics and randomized procedure to tackle the complexity of the TSP. Several classes of these have been developed, including *tabu search, iterated search, iterative improvement, simulated annealing, ant colony optimization as well as greedy and genetic algorithms*. For an in-depth treatment of topics related to the TSP, the reader is referred to APPLEGATE ET AL. (2006).

2.2.3.3 Application of Heuristic Approaches to the Sensor Planning Problem

The majority of the viewpoint selection schemes introduced in section 2.2.2 apply search heuristics to connect viewpoints to form a near-optimal scan path. The researchers involved in these studies have come to the conclusion that most of the search heuristics will give satisfying results within an acceptable time frame. SCOTT ET AL. (2001) even believe that at path planning stage, computational resources is better devoted to generating high-quality viewpoints than to the calculation of scan

paths. In other words, they believe that the most difficult challenge of the sensor planning is framing the problem, not necessarily solving it by route search.

To give a complete overview of path computation schemes, it must be mentioned not be omitted that a small marginal number of works have chosen non-heuristic approaches to tackle this issue. In SHU & XI (1999), the scan paths are constructed by connecting straight-line segments along the slicing plane with step-over segments to cover the entire WP surface. Here, the end point of the first straight line is connected to the starting point of the closest step-over segment. Next, the end point of this step-over segment is linked to the starting point of the next straight line, and so on. The result is a zigzag path over the WP surface. As shown in figure 2.7, the algorithms developed by ELMARAGHY & YANG (2003) also operate similarly. However, zigzag paths have not experienced much acceptance in path planning due to their lower efficiency and longer travelling time (SCOTT ET AL. 2001).

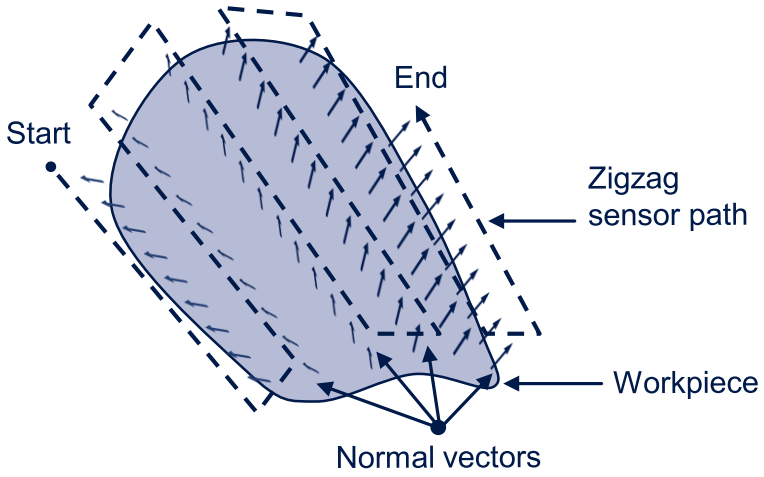


Figure 2.7: Zigzag path for complete surface covering

2.2.4 Derivation of Scan Trajectories

2.2.4.1 Path vs. Trajectory

Before proceeding with this section, the definition of the terms "trajectory" and "path" which are often erroneously used as synonyms, needs to be clarified. A path as illustrated in figure 2.8 denotes a locus of points connecting two or more destina–tions. A trajectory, in contrast, is a path on which a time law has been specified (NNAJI 1993 p. 154). In practice, computing scan trajectories comes down to ensuring that the sensor head can be steered along the pre-computed path by the robot with a given velocity. This processing step is assured by a trajectory planner which takes the pre-computed path and the manipulator's kinodynamic constraints as inputs (DUNN & OLAGUE 2004). Kinodynamic constraints here entail kinematic and

dynamic restrictions on the manipulator movements. Kinematic constraints may include, but are not restricted to, joint limits and obstacles limiting the freedom of movement of the robot. Dynamic restrictions are those which primarily govern the dynamics laws of the robot motion. They include, but are not limited to, bounds on the velocity, the acceleration and the applied motor forces as well as singularities.

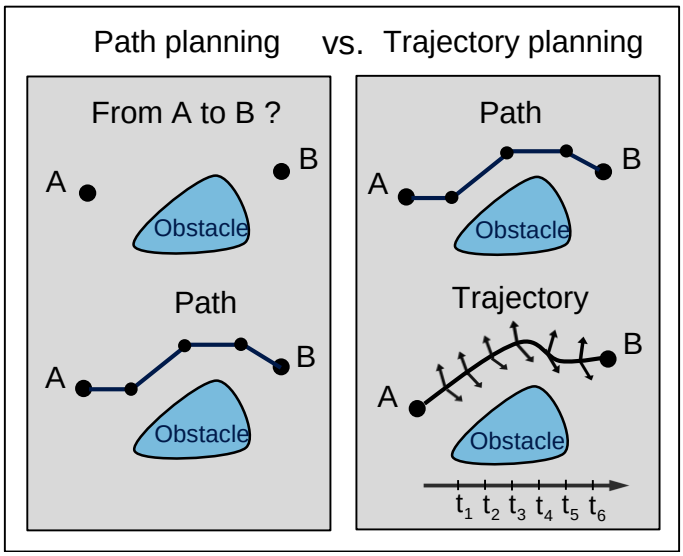


Figure 2.8: Illustrative explanation of the terms "path" and "Trajectory" (derived from MUNZERT 2008 p. 41)

2.2.4.2 Considering the Manipulator Constraints

Sensor planning has long been considered a purely geometrical challenge. This is illustrated by the fact that the sensor planners examined in several studies were designed only to produce paths specifying the sequence of the sensor's positions and orientations instead of trajectories (RENTON ET AL. 1999, CALLIERI ET AL. 2004). Those paths are then assumed to be drivable by a robot—which is not always the case due to limitations on torque available from IR's actuators. The aspect of how fast the sensor is to progress along the path is not considered at all. This convenient omission of the IR characteristics at planning time squares with the belief that sensor planning becomes increasingly easier if this is the case (SCOTT 2009).

Meanwhile, the importance of incorporating robot kinodynamic constraints at sensor planning stage has been acknowledged. One then speaks of "performance-oriented sensor planning". Such planning methods have already been successfully implemented for FISs equipped with simple positioning mechanisms, e.g. spherical robots (TRIGGS & LAUGIER 1995), translational stage (PAPADOPOULOS-ORFANOS & SCHMITT 1997, LORIOT ET AL. 2007) and Cartesian robots (DERIGENT ET AL. 2007). Even though they have all claimed that their work can be applied to all kinds of

kinematics with minor changes, it is legitimate to doubt whether that is the case, in light of the simplifying assumptions adopted therein. Indeed, a robot manipulator with six-axis revolute joints makes it very hard to predict possible violations of its physical limitations. Hence, these limitations have to be explicitly considered in order to make sure that the sensor trajectories comply with them.

2.3 Programming of Industrial Robots

2.3.1 The Need for Easing the Use of Industrial Robotic Systems

"Old industrialized nations" are experiencing fierce competition from emerging countries with different salary structures, social organization, and social protection. Beyond that, an impending aging issue characterized by the transition towards a much older population can be observed in the former country group (ZÄH & PRASCH 2007). To remain competitive, these nations need significant increases in industrial efficiency—precisely the reason what IRs are made for (GROOVER 2008 pp. 229–231, ALMEIDA ET AL. 2006 pp. 515–528).

Just as the small and medium-sized enterprise (SME) is the most flexible type of company, IRs are the most flexible type of production equipment (PIRES ET AL. 2005). Nevertheless, due to the high cost of ownership associated with running them (NAUMANN ET AL. 2006), especially in the case of small batch production, robotic applications are still not a commodity in SMEs (JOHANSSON ET AL. 2004; BRECHER ET AL. 2006). Typical expenses in addition to of the main purchasing costs include programming and running costs, maintenance, and spare parts. Of these, programming costs represent the highest barrier faced by SMEs, as they currently account for the biggest share of expenditure (UHLMANN ET AL. 2008, NETO ET AL. 2010a). However, because precisely SMEs are purportedly the real giants of the European economy, it is obvious that this hurdle has to be cleared in order to widely open this unsaturated market to IRs. Indeed, in the European Union, SMEs account for 98% of an estimated 23 million enterprises, with 93% having fewer than 10 employees (EUROCOM 2005). Hence one should be quick to realize that robotic systems have to be better adapted to SME needs in order to support these companies sustainably in their competition against low-wage countries (Dressier ET AL. 2007, SOM 2010). As acknowledged by many scientists, a clear path toward hitting this target lies in the reduction of the programming and (re)configuration efforts applied to robotic systems (WESTKÄMPER ET AL. 1998, SCHRAFT & MEYER 2006, SOLLER & HENRICH 2009, KRÜGER ET AL. 2011).

2.3.2 Industrial Robot Programming Concepts

2.3.2.1 Background

Robot programming aims at translating a production task into a sequence of robot motions in an order that will accomplish a desired task, while considering manufacturing resources and constraints (NNAJI 1993 p. 3). A task is defined here as a partially ordered set of technological operations. Robot programming methods can be categorized using several taxonomies. A common taxonomy makes a distinction between online and offline programming methods with regard to whether the programming is performed using the physical IR or a virtual representation of it (WECK & BRECHER 2005 p. 572). However, since motion specifications portray the most obvious aspect of robot programming, a classification emphasizing the degree of abstraction of the programming process would appear to be meaningful with respect to the topic of this dissertation. Thus, following BIGGS (2007 pp. 11–27) and THOMAS (2008 pp. 7–9), the field of robot programming is roughly subdivided into conventional and automatic robot programming.

2.3.2.2 Conventional Robot Programming

As shown in figure 2.9, conventional robot programming methods are characterized by the fact that the user has to create the robot program by hand at a low abstraction level (BRECHER ET AL. 2010).

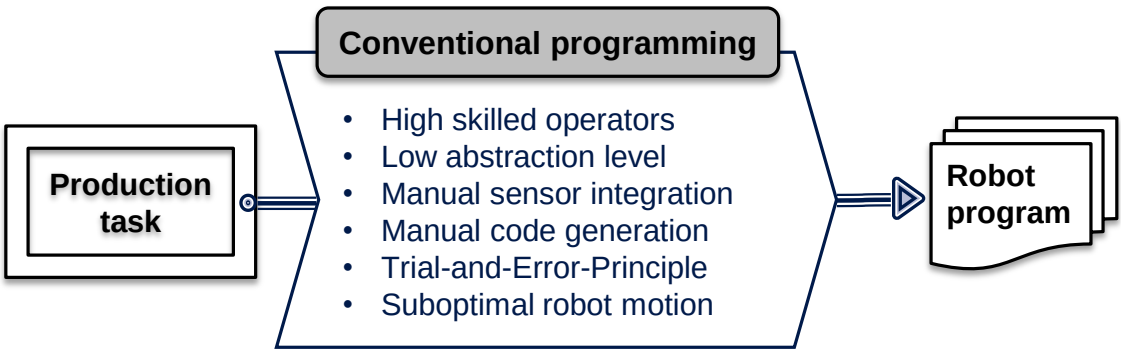


Figure 2.9: Conventional programming

The programming is usually carried out using human machine interfaces such as teach-panel or computer-based input devices. To program the robot movement, operators have no choice but to rely on trial-and-error methods (LUEDEMANN-RAVIT 2005 p. 1). Error-prone programs have then to be debugged by repeatedly re-editing, downloading, and executing the program. Furthermore, sensor logic also has to be manually integrated by the system programmer. All these stages require

specialists who are comfortable with complex programming languages and with the integration of sensor technologies, which are skills not usually possessed by employees on the shop floor (HOLLMANN 2009).

Conventional programming methods can be roughly subdivided into robot level programming, also known as Teach-In, and simulation-based programming. Today, Teach-In is still the most widely used programming method employed in the industry (LAST & HESSELBACH 2006, BRANDENBURG & DILTHEY 2006 p. 288). This is quite astonishing in view of the fact that robot-level programming is conducted directly on the physical robot, meaning that it has to be temporally withdrawn from manufacturing tasks. Associated with this is a standstill of production facilities (NOF 1999 p. 756). Teach-In programming techniques can thus be very costly since the programming time in which the production facilities cannot be productively used may, in some cases, last days or even weeks.

Although simulation-based programming can be performed off-line and requires less technical skill than robot-level programming, the user still has the responsibility of designing the desired trajectories in term of reference frames, position and orientation of the tool as well as the traversing sequence (CARVALHO 1997 p. 18). Another drawback of simulation-based programming lies in the deviation between the real and simulated worlds, which requires costly calibration equipment or some re-teaching activities to adjust off-line generated programs (BRECHER ET AL. 2010).

2.3.2.3 Automatic Programming

As explained in the previous section, manual programming methods force the programmer to think at the level of joint and Cartesian coordinates, trajectories, and collisions or, even more complicated, in terms of robot-specific instructions. According to PRINZ ET AL. (1996), KUGELMANN (1999 p. 5) and (MEYNARD 2000 p. 21) it would be better if system programmers could think instead at a more abstract level with which they are comfortable, leaving the machine to fill in the details. This is precisely what the automatic programming methods strive to achieve. Here, the programmer does not have direct control over the robot program code generated. He just limits himself to abstractedly defining the manufacturing task which is then automatically translated into trajectories executable by the robotic system. from this, the benefits of automatically programming IRs could not be clearer: short programming and ramp-up time, brief changeover phases, efficient process cycle time, reduction of energy consumption through optimized trajectories, decrease of training costs, to name only a few. Figure 2.10 illustrates the advantages of automatic robot programming and especially the gain in time with respect to conventional robot programming.

Many paradigms have arisen in the area of automatic robot programming as people continue looking for ways of reducing the effort of programming robot. An exhaustive survey of robot programming systems can be found in BIGGS (2007 pp. 12–27) and VOGL (2008 pp. 11–24). As a full review of automatic programming methods is beyond the scope of this dissertation, only the most relevant of them, the task-level programming, will be given a closer look.

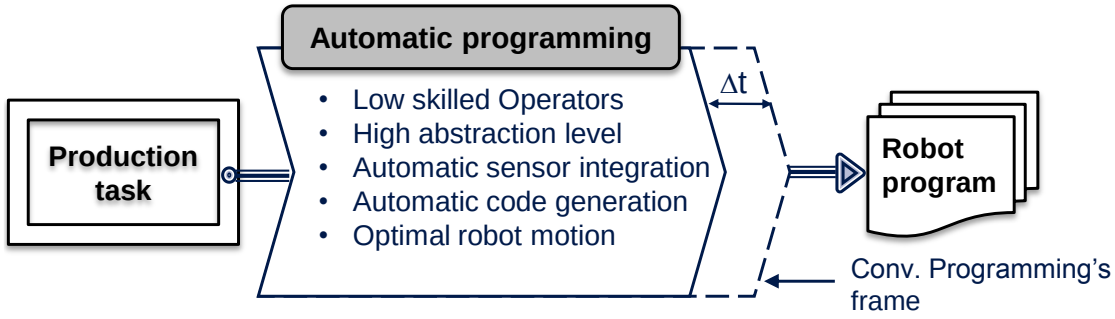


Figure 2.10: Automatic robot programming

2.3.3 Task-Oriented Programming: A Closer Look

2.3.3.1 Definition

The idea behind task-oriented robot programming (ToP) is to hide all the machine details and let the user specify an application at a higher level of abstraction (LOZANO-PÉREZ & BROOKS 1986). This enables the user to specify the desired goals of the programming tasks without defining every movement of the robot. The emphasis here is on defining "what" to do instead of "how" to do it. Most of the time, the user is limited to inputting abstract instructions such as “insert PEG in HOLE” (LOZANO-PÉREZ & WINSTON 1977) or “WELD SEAM” (REINHART ET AL. 2008). From this, the programming system then automatically generates program codes that allow the robotic system to perform the production task. ToP has been recognized as the way to go in the attempt to ease the robot programming effort (THOMAS 2008 p. 11). Furthermore, its effectiveness has recently been proved through various experiments reported in NETO ET AL. (2010A). For instance, an experimental comparison between a ToP system and manipulator-level programming system conducted there has brought the insight that the former are much more intuitive and easy to use than the latter. Moreover, ToP methods exhibit shorter learning curves allowing novice robot operators to generate robot programs in just a few minutes. The experiment involved two test subjects, with basic skills in CAD and 3D modeling but who had never worked with an IR before.

In order to anticipate any potential confusion between task-oriented and simulation-

based programming methods, their fundamental difference should be clarified here: in contrast to ToP, simulation-based programming approache still requires the operator to have advanced CAD, programming and robotic skills. These technical skills are necessary to learn the language of the simulation system and perform the planning of the robot motion as it has to be achieved wholly manually (SICILIANO & KHATIB 2008 p. 979).

2.3.3.2 Architecture of Task-Oriented Programming Systems

Figure 2.11 depicts the generic architecture of ToP systems according to NNAJI (1993 p. 67–73). From this architecture, it is apparent that a ToP must at least incorporate:

- an environment modeling module to capture the manipulator's virtual model;
- a task-specification module to abstractly specify the task at hand;
- a robot motion generation module that automatically synthesizes the movement of the IR;
- a robot interface module to connect the ToP system with the physical robot.

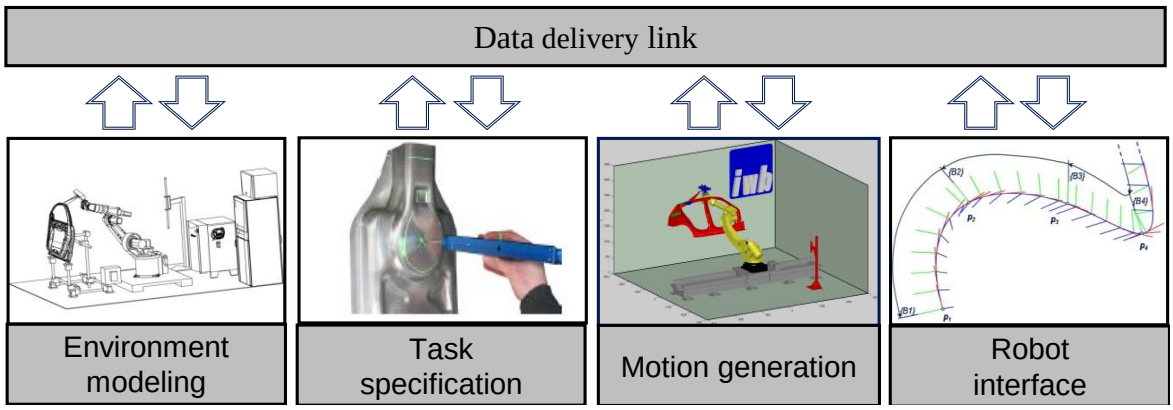


Figure 2.11: Generic architecture of ToP systems

Environment modeling module

ToP requires a representation of all the objects of the robot workcell (e.g. robots, tools, WPs, jigs) as a knowledge base to reason on (LOZANO-PÉREZ & BROOKS 1986, VOGL 2008 p. 23). The state of these environment objects is represented in a world model. Extracting the geometrical description of the workcell component directly from a CAD system has proved a convenient way to geometrically model robots' working environment. The world model is not only restricted to the geometrical description of the workcell, but also entails the mathematical description of the robot physical structure.

Task specification module

The specification of the task to be carried out by the robotic system is accomplished through a communication mechanism between the operator and the programming system. Mostly, high-level textual languages have been used, whose commands exhibit structures very close to the natural language of the operator. The user-friendliness of the task specification module is of major importance, as it can significantly bias the programming system acceptance (MEYNARD 2001 p. 27) or rejection (FRIEDRICH 2010 p. 34). However, high-level abstraction human-robot interfaces are still not a commodity in ToP systems (NETO ET AL. 2010B).

Motion generation module

This module is in charge of analyzing and breaking down the task specification into elementary tasks. While the environment modeling and the task specification step are to be carried out by the programmer, this step is automatically performed by the programming system. The central functionality of this module is to dissect user-entered specifications and generate adequate robot movements to carry out the task within the constraints imposed by the world model. The task-planner itself may contain several algorithms that might be termed "intelligent" and handle solving sub-problems such as global and local motion planning, collision detection and avoidance (NOF 1999 p. 355).

Robot interface module

The robot interface element of the system creates the link between the virtual computational environment and the real world. It finalizes the programming job by converting the robot movement generated in the previous phase into native robot commands that are then downloaded to the physical robot controller for execution. This interface may also be equipped with the functionality to remotely control the robot or to monitor the robot system at execution time.

2.3.3.3 Drawback of ToP systems

Robot environment modeling inevitably involves discrepancies between the virtual and the real world models (HIRZINGER ET AL. 1994). These discrepancies may have several origins. For instance, the robot may not be built exactly according to the design specifications captured in the CAD model. Additionally, the tool mounting position and orientation may not be exactly determined, or the spatial arrangement in the environment model may not correspond exactly to that of the real world. Whatever the origin, those discrepancies normally cause robot programs generated in ToP to work improperly. There are several more or less complex approaches to achieving the best possible agreement between simulation and reality. One of them

is to adjust the generated program by re-editing it in the real workcell. Another solution may consist in either applying complex clamping devices to enforce the workcell spatial arrangement or by devoting much more attention to modeling activities in order to make them more accurate. However, from an economic point of view, none of these approaches are appropriate for small series production (JOHANSSON ET AL. 2004). An economically favorable way to overcome this dilemma and assure that the IR stays on track despite the environment model uncertainties is the active use of sensor data (MILBERG ET AL. 1997). Depending on the application at hand, this may take the form of optical, tactile, ultrasonic sensors or through-the-arc-sensing (CARVALHO 1997 pp. 31–32).

2.3.3.4 Existing Works

Much of the earlier efforts in the field of ToP have been concentrated on the development of systems for automated assembling: (LOZANO-PEREZ 1983, PERTIN 1987, KAPITANOVSKY & MAIMON 1993, LUETH & LAENGLE 1994, MYERS 1999, KUGELMANN 1999). These studies draw on the assumption that each assembly task can be deconstructed into a restricted set of atomic steps, called manipulation task primitives, for which a set of motion sequences is then pre-programmed. Subsequently, once the assembly task is specified, programming the manipulator is reduced to choosing and adequately sequencing some of those atomic steps. As the methodologies developed in these studies are specifically tailored to assembly tasks, they are not transferable to FISs.

Intensive work has also been done recently to provide methods and programming support for other application fields. In PIRES ET AL. (2004), a series of studies was conducted to automatically synthesize robot welding trajectories using the CAD Data eXchange Format (DXF). Unfortunately, the programming system was tested using only linear welds and the user still needs to define the approach and escape paths. In REINHART ET AL. (2008) a ToP system for remote laser welding is presented. The effectiveness of the welding trajectories generated by the system led to a reduction of cycle time of more than 30%. However, the welding task has to be defined by means of an augmented reality-based user interface, calling for additional hardware such as tracking cameras. KIM (2004) and WU ET AL. (2009) introduced methods for generating three dimensional trajectories for a robotic adhesive spray system for shoe outsoles and uppers are presented. FENG-YUN & TIANSHENG (2005) introduce an automatic robot path generator for the polishing process, where the cutter location data is generated by the postprocessor of a CAD software. Systems to automatically program coating and painting robots are presented by CHEN ET AL. (2005) and BI & LANG (2007).

2.3.3.5 Summary

ToP has improved robot technology as much as object-oriented programming has improved software engineering (CEDERBERG 2004 p. 39). Nevertheless, there are still tremendous obstacles that must be overcome to find general solutions to the problem of automatic programming. One of them, and perhaps the most prominent, is the difficulty of developing generalized mechanisms to unambiguously translate complex real-world scenarios into task description (DA COSTA ET AL. 1992). These difficulties clearly show that it is not realistic to put effort into finding a generic task-oriented approach valid for the vast area of robotic applications. But still, even if general mechanisms cannot be found, particular applications can and will benefit from this methodology. FIS-based QI applications may be one of them.

2.4 Recapitulation and Discussion

The first part of this chapter was devoted to reviewing literature dealing with sensor planning. The gain in knowledge acquired thereby can be summarized as follows:

- Current FIS-related sensor planning procedures are error-prone, time-consuming and needlessly tie up expensive metrology equipment.
- Automating the sensor planning constitutes an effective step towards reducing the time, cost and manual labor currently encountered while programming of FISs.
- Possessing and understanding the shape of the object being measured allows a more complex and specific scan plan to be generated.
- Multi-staged viewpoint selection schemes are appealing approaches because they can draw from different techniques for each processing stage.
- In performance-oriented sensor planning, it is not sufficient to consider only the imaging constraints which characterize the image quality. Performances of the sensor carrier (i.e. IR) have to be taken into account too.
- Existing sensor planning methods are tailored to trivial kinematic system like translational and rotational stages and CMM.
- Most of the existing sensor planning methods addresses the issue of generating sensor scan trajectories as a purely geometrical problem. Accordingly, sensor planners which operate under both kinematic and dynamic constraints of non-trivial IR kinematics are still lacking.

The second part of this chapter in contrast was concerned with the topic of robot programming, though emphasis was placed on automatic programming methods. The main points discussed there can be summarized as follows:

- In order to assist SMEs in their quest for competitiveness and productivity, reducing the robot programming effort is the right way forward.
- Intuitive human interfaces have to be associated with ToP systems to reduce the complexity of task description.
- There are many initiatives for the development of ToP systems, but so far only the following manufacturing processes have been extensively targeted: assembly, deburring, and painting, as well as arc and remote laser welding.
- ToP systems have difficulties in coping with the geometrical mismatch between the virtual and real robot worlds. Hence they require time-consuming calibration procedures to adjust the simulation results.
- There is a dearth of dedicated support for combating the programming complexity of FISs. APS dedicated to other manufacturing processes cannot be easily adapted to fit the requirements of FISs.

The above statements and shortcomings expressly underline that there is still a strong incentive to pursue the easing of the programming labor applied to FISs. Accordingly, this research should be understood as an attempt to close the gap between FISs and other robotic applications like painting, , deburring and remote laser welding, where practicable solutions to alleviate the programming burden have already been implemented. The solution devised here consists of the blueprint of a methodology aimed at reducing the effort currently associated with programming FISs. The methodology is specially tailored for high-variant, small batch production, as its impact is expected to be the highest there (DENKENA ET AL. 2005).

3 Conceptualization of the New Methodology

3.1 General Description

Based on the insight and the findings of the discussion in the previous chapters, this chapter proposes a new programming methodology geared towards minimizing the level of robotic and metrology knowledge required to program FISs. Figure 3.1 shows a flow diagram of this programming methodology, comprising the following five activities:

- CAD-based extraction of the WP design features;
- Feature-based description of the digitization task;
- Automated generation of the scan trajectories, which allows the sensor to capture quality-relevant features;
- Automatic synthetization of suitable robot movements to steer the sensor along the generated scan trajectories;
- The online adaptation of the scan trajectories at execution time to cope with the mismatch between the real and virtual worlds.

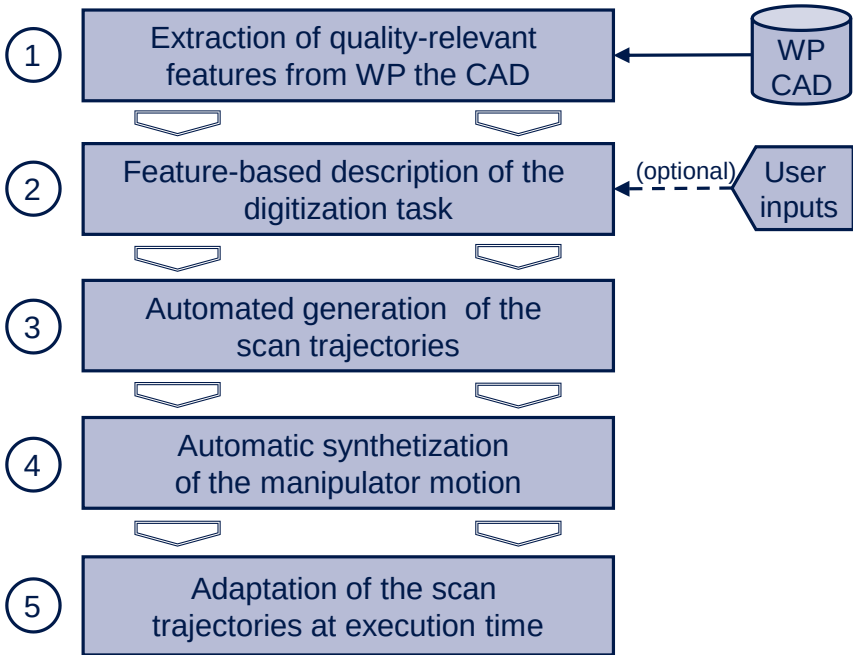


Figure 3.1: Flowchart of the programming methodology

The idea behind this concept is to limit the user interaction to no more than the specification of the digitization task at a high level of abstraction, leaving it to the programming system to decide on how to execute the task. Hence, the methodology is characterized by a approach that combines the benefits of "performance-oriented"

sensor planning with those of automatic robot programming to unburden the system programmer of complex details.

3.2 Automatic Programming System

To furnish evidence of the methodology's achievement potential, it is implemented prototypically in the form of an APS. Following the maxim of "divide et impera,"[5] the architecture of the APS is broken down into four modules as depicted in Figure 3.2. Each of these modules encapsulates a clearly delimited aspect of the methodology introduced above. In this way, the APS's modules are lent a high degree of reusability that allows them to be easily redeployed in other ToP scenarios. In the subsequent chapters, these modules will be developed independently and then plugged together.

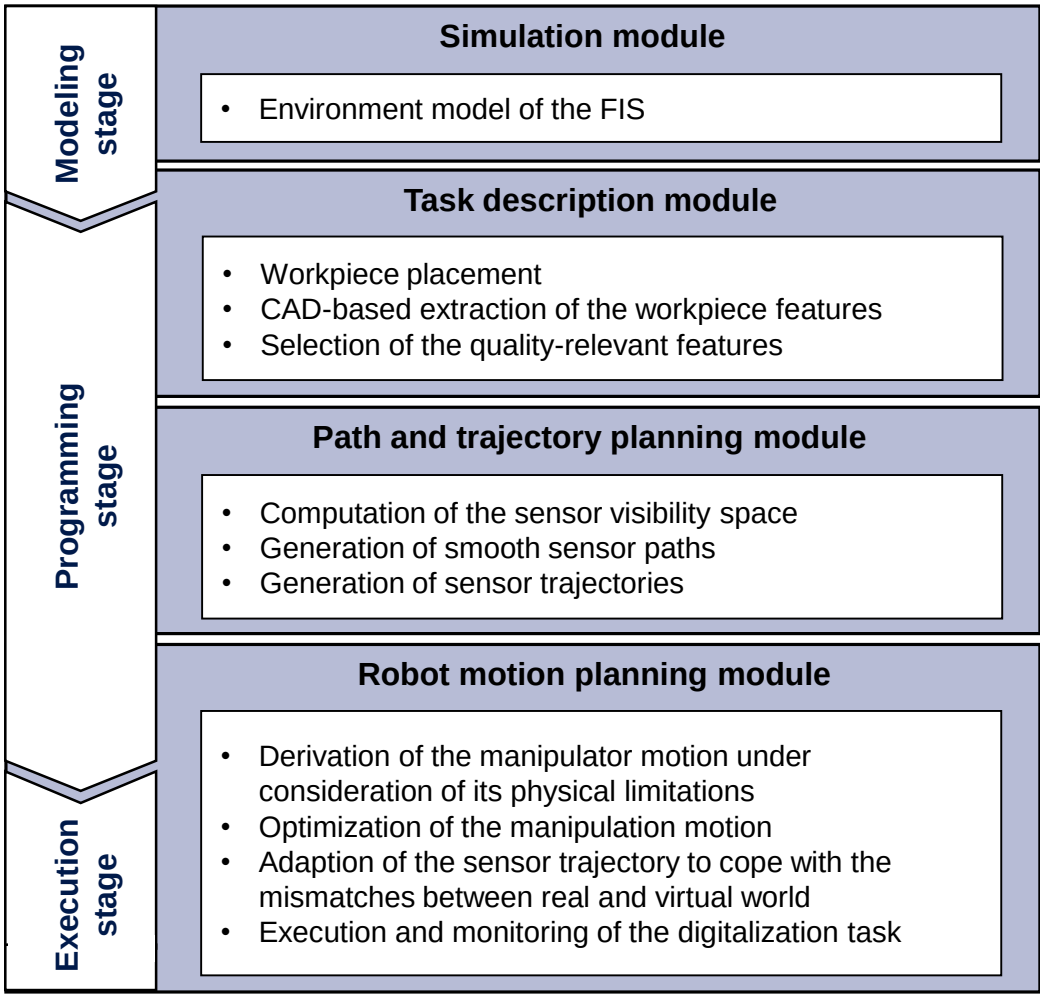


Figure 3.2: Components architecture of the APS

[5] Algorithm paradigm aimed at solving a complex problem by splitting it into smaller sub-problems. (English: divide and conquer)

3.3 General Description of the Modules

In order to shed more light into the proposed programming methodology, each module of the APS is briefly described in this section.

Simulation module

Due to its task-oriented nature, the APS requires a comprehensive database containing not only a world model, but also metrology-related knowledge. Therefore, the first module of the APS is a simulation module that accommodates these requirements. Additionally, this module is intended to provide further services to the other modules, such as collision query and graphical animation.

Task description module

In this module, the FIS operator describes the digitization task at hand based on the WP features. To this end, using the WP CAD model, feature contours are automatically extracted as specified by the design engineer, and displayed. When needed, the module enables the user to decide which features are relevant for the inspection task. As a default, all of the extracted features are considered. Since the relative location of the WP with respect to the FIS determines whether its AoI can be captured or not, this module also provides the means by which the workcell layout can be set up to address this issue.

Scan path and trajectory planning module

The feature contour lines output by the previous module are utilized to automatically generate scan trajectories that allow the sensor to collect the quality-relevant features. The module starts by computing a sequence of collision-free sensor positions and orientations allowing for robust coverage of the feature contours. The view space for searching the scan paths is reduced in size and dimensionality by the application of the digitization constraints. Secondly, the scan paths are smoothed and then augmented with a time law that turns scan paths into scan trajectories. It is important to point out that, at this stage, the constraints related to the manipulator are neglected in order to leave more scope for the sensor planning algorithms. This decoupled approach is found by SHILLER & DUBOWSKY (1991), and more recently by VERSCHEURE ET AL. (2009) to be more effective than approaches geared towards solving the motion planning problem in one stroke.

Motion planning module

This module is assigned the task of deriving adequate motion of the manipulator to guide the sensor in accordance with the scan trajectories issued by the preceding module. This module accounts for the physical restrictions of the manipulator in term of kinematic constraints. Singularity issues and the problem of minimizing the

configuration changes are addressed. Additionally, it is shown that the shape of the scan trajectories does not violate the FIS's dynamic restrictions. To reverse the deviation caused by the mismatch between the virtual and real worlds, this module implements a visual servoing system that monitors the sensor readings in real-time and adjusts the robot motion so that the FIS remains on track.

3.4 Methodology Rationale

Regarding the feature-based task description

The feature-based task description method implemented in this study rests on the observation that artifacts such as edges, corners, slots and pins are critical to producing high-quality sheet metal parts. Hence, dimensional QI of these parts can be expected to consist of checking these critical features against their nominal specifications. It is therefore legitimate to assume that the inspection task, and by extension the digitizing task, can adequately be described using the contour lines of these features. Admittedly, capturing the complete WP surface may be required in some cases, e.g. in the early phase of product development. For such cases, it should be sufficient to scan the entire surface of the WP using a zigzag motion, as shown in figure 3.3. To prove this hypothesis, a rudimentary zigzag path generator was implemented and made available within the APS.

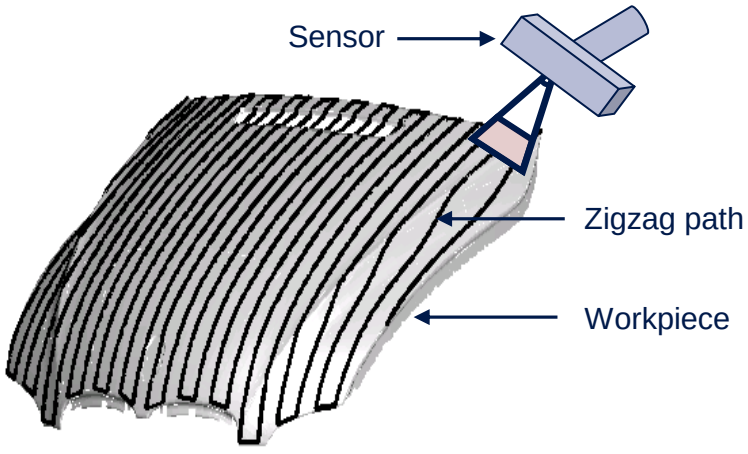


Figure 3.3: Zigzag sensor path for full surface coverage

This planner implements the algorithm introduced in (SHU & XI 1999) to produce line segments that are then connected using a method similar to the one described in ELMARAGHY & YANG (2003). Once the zigzag path is generated, it is handed over to the scan path and trajectory planning module as if it were a feature contour line. In this way, the new programming methodology presented in this research can also be deployed for the complete digitization of WPs. However, since the strength of

FISs does not lie in full-surface coverage but rather in gathering specific AoI on a WP surface, the emphasis of the new programming methodology logically is focused on the latter aspect.

Regarding the programming effort

Within the new programming framework detailed in section 3.3, FIS programming is achieved via three phases: a modeling, a programming, and an execution phase (see the white vertical bar on figure 3.2). In accordance with BRECHER ET AL. (2006) these phases temporally separate non-recurring (modeling) and the recurring (programming and execution) activities dealt with during the FIS life cycle. Because the modeling phase requires deeper technical knowledge, it obviously has to be performed by a highly skilled specialist. Should the programming methodology still be viewed as relieving the human programming effort? The answer to this question is a definite yes. Indeed, the programming and execution phases that represent the daily operations to be performed on the FIS in a batch production environment are almost fully automated. The only human intervention required is the selection of some of the WP's features in the task description module. This step requires only a point-and-click on the 3D graphic model displayed in the Graphical User Interface (GUI), which can easily be carried out by operators with limited exposure to robotics and metrology knowledge.

3.5 Assumption and Convention

In the treatment of the problem addressed in this thesis, the following assumptions are made:

- The relative size of the WP and the manipulator are compatible, i.e., the robot is large enough to reach every corner of a contour line to be digitized.
- No constraints are imposed on the length of the manipulator links as long as the first assumption is fulfilled.
- It is assumed that the sensor integration—the task of getting the sensor and the manipulator to recognize each other—has been performed prior to the system programming phase. For instance, during the commissioning phase.
- The term "sensor" refers exclusively to LSSs, unless otherwise specified.
- It is assumed that the IR carrying the sensor is a serial robot with six revolute joints of which the last three form a spherical wrist hand. That is, the three last joints intersect at one point, the Wrist Center Point (WCP). Throughout this thesis, the terms "IR", "robot" or "manipulator" will be used interchangeably to refer to this robot type.
- The handling of the metrology interfaces in terms of data registration and

alignment will not be addressed as it is far beyond the scope of this work.

- By the same token, the focus of this research does not include topics related to FISs, such as absolute calibration, temperature compensation data registration, and CAD-to-part comparison.

4 Simulation Module

4.1 Overview and Module Structure

Due to its task-oriented nature, the new programming methodology necessitates an environment model of the FIS's workcell. In an effort to fulfill this requirement, a simulation module will be elaborated in this chapter. It is intended to serve as a knowledge base upon which algorithms embedded in the upcoming modules will rely on. Figure 4.1 outlines the main components of the module while simultaneously representing a navigation map of the core topics addressed in this chapter.

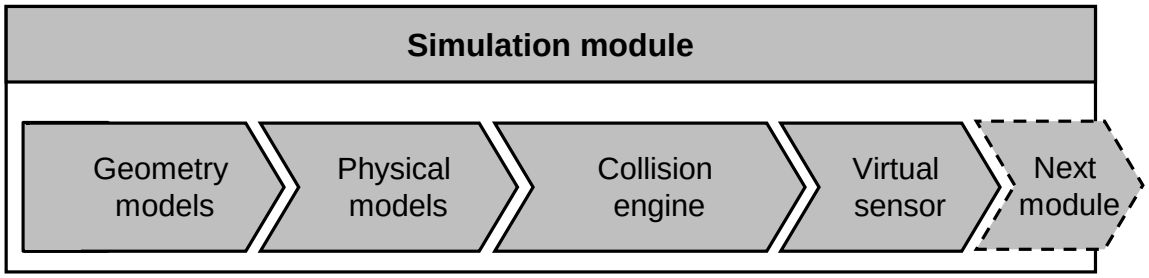


Figure 4.1: Module overview

4.2 Geometry Model of the FIS Workcell

4.2.1 CAD Interface

The purpose of the geometry model is to reproduce the volumetric shape of the FIS working environment. Such a geometry model is useful for sensor and motion planning activities as well as for visualization purposes. In the framework of the new programming methodology, the geometry model of the FIS workcell is intended to be obtained by importing a 3D drawing of each of the workcell components into the simulation module. Unfortunately, numerous 3D modeling techniques exist, as well as a large variety of vendor specific data structure for the storage of 3D drawings. To complicate matters, some of these data structure are undisclosed to the general public s. On this account, the Surface Tessellation Language (STL) was favored as the interfacing data format to the simulation module. Originally developed for stereolithography machines[6], STL has become a de facto standard for the digital exchange of geometrical shape information among Computer-Aided engineering technologies (CAx) and systems (LEFEBVRE & LAUWERS 2004). One of

[6] Stereolithography machines—sometimes also referred to as 3D printing machines—are rapid prototyping apparatuses that can build volumetric shapes a layer at a time.

the most advantageous features of the STL format is that almost all CAD systems can output it or convert existing proprietary file formats into it.

The STL format replicates the shape of a 3D solid object by approximating its shell with a mesh consisting of planar, oriented triangle (see figure 4.2). Each triangle is described by a unit outward normal and three points listed in counterclockwise order representing the vertices of the facet. The aspect ratio and orientation of individual facets is governed by the surface curvature. A file parser[7] to decrypt the syntax of the STL file and store the CAD data in an appropriate data structure was implemented in this module. The data structure captures the normal vectors' data and the vertices' data of the mesh on a per facet basis. To accelerate the traversal and processing of the mesh, additional arrays containing connectivity information between vertices and adjacent triangle are also part of the data structure.

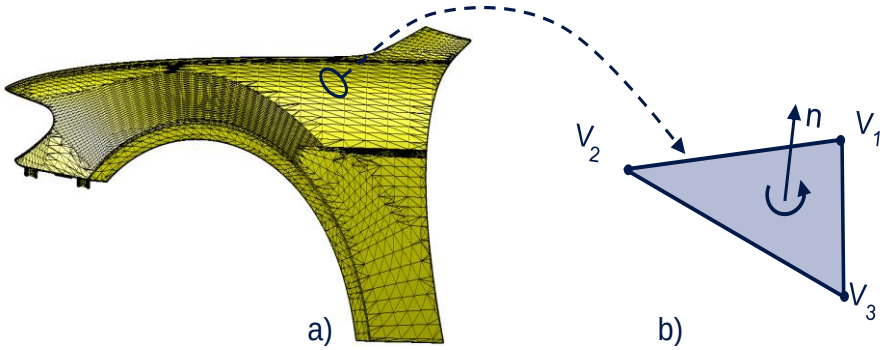


Figure 4.2: a) Meshed 3D model of a sheet metal WP; b) Illustration of a STL triangular facet

4.2.2 Spatial Relationship in the Workcell

As soon as the geometry models of the components are imported, a right-handed coordinate frame is attached to each of them to keep them traceable. As depicted in figure 4.3, each of these frames describes the position and orientation of the corresponding object it is attached to with respect to a fixed reference coordinate system F_{WC}. Each frame is mathematically characterized by a *homogeneous transforms* that expresses the spatial relationship between a reference frame and a remote frame in a 4x4 matrix as follow:

$$F = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

[7] A software routine that walks through the STL file, identifies all relevant data, stores them in an appropriate data structure and passes it back to the application that invoked it.

The matrix entries r_{ij} $i,j \in [1\ 3]$ represent a 3x3 rotational sub-matrix expressing the change in orientation of the remote frame with respect to the reference frame axes. The entries p_x, p_y and p_z represent a 3x1 vector which captures the position of the remote frame with respect to the reference frame. Nevertheless, an operator who wishes to enter the location of the workcell components will find it very laborious to input several 4x4 homogenous matrices. For this reason, frames are also represented throughout this research by the following more compact form

$$F = [x\ y\ z\ \gamma\ \beta\ \alpha] \tag{4.2}$$

where x, y and z represent the frame's position and γ, β and α label a ZYX-Euler angle set describing the frame orientation with respect to the world CS F_{WC}.

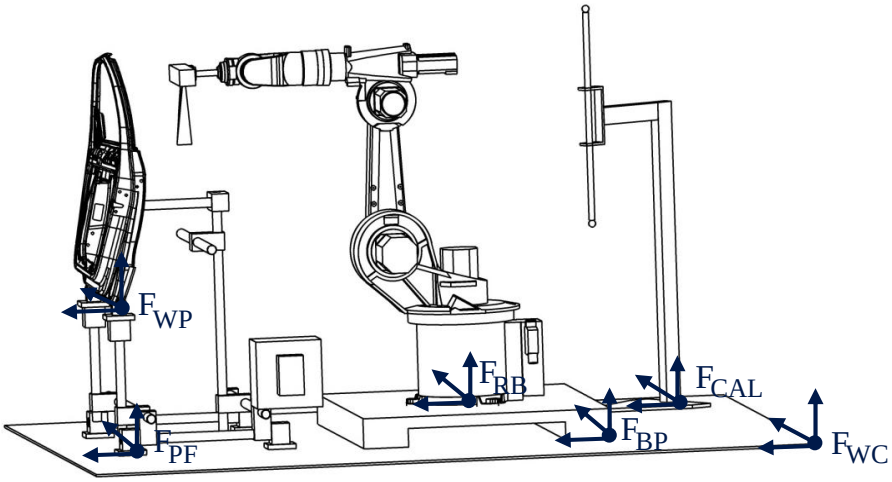


Figure 4.3: Coordinate frames used to describe the spatial arrangement of the virtual FIS model

4.3 Physical Modeling of the Manipulator

4.3.1 The FISs' Kinematic Chain

From a mechanical point of view, the FIS may be thought of as a serial chain of rigid bodies or links, where neighboring link pairs are interconnected by revolute joints. One end of the chain is constrained at the base, which should be fixed at the location where the robot is attached to its basement plate. Beginning from this base link, each link supports following links mechanically. The last, outermost link is the sensor probe. Such an arrangement of actuators and linkages is called an open serial kinematic (CRAIG 2005 p. 85).

To mathematically capture the FIS's kinematic chain, the kinematic formalism by

DENAVIT & HARTENBERG (1955) is applied. This formalism, also known as the DH convention, rules that each articulated link has to be attached to a frame laid out in a specific manner. A detailed description of the DH convention is omitted here for brevity. However, it is important to keep in mind that the choice of the frames may not be unique for the same kinematic chain. Figure 4.4 shows the frame assignment to the FIS according to the DH convention. The FIS's links are numbered from 0 to 6. The base link is *link0* and the outermost, *link6,* is coupled to the sensor. The frame attached to the i^{th} link is labeled $F_i(x_i\ y_i\ z_i)$. The FIS working point, or Teach Center Point (TCP), is located in the middle of the sensor measuring range. To this TCP, is associated a frame, which for the time being will be labeled F_{TCP}.

Referring to figure 4.4, the $F_0(x_0\ y_0\ z_0)$ attached to *link0* can be transformed into the frame $F_1(x_1\ y_1\ z_{1)}$ attached to *link1* by the following successive homogenous transformations:

- a rotation about the axis z_0 by an angle θ_1 to make vector x_0 parallel to z_1
- Translation along the axis z_0 of a distance d_1 to make vector x_0 and x_1 collinear
- Translation along the axis x_0 of a distance a_1 to make the origin of F_0 congruent with the origin of F_1
- Rotation by an angle of α_1 to align axis z_0 with axis z_1

The operation above results in four transformation matrices which, when post-multiplied, yield the composite homogenous matrix

$$^0T_1 = Rot(\theta_1) \times Trans(d_1) \times Trans(a_1) \times Rot(\alpha_0) \qquad (4.3)$$

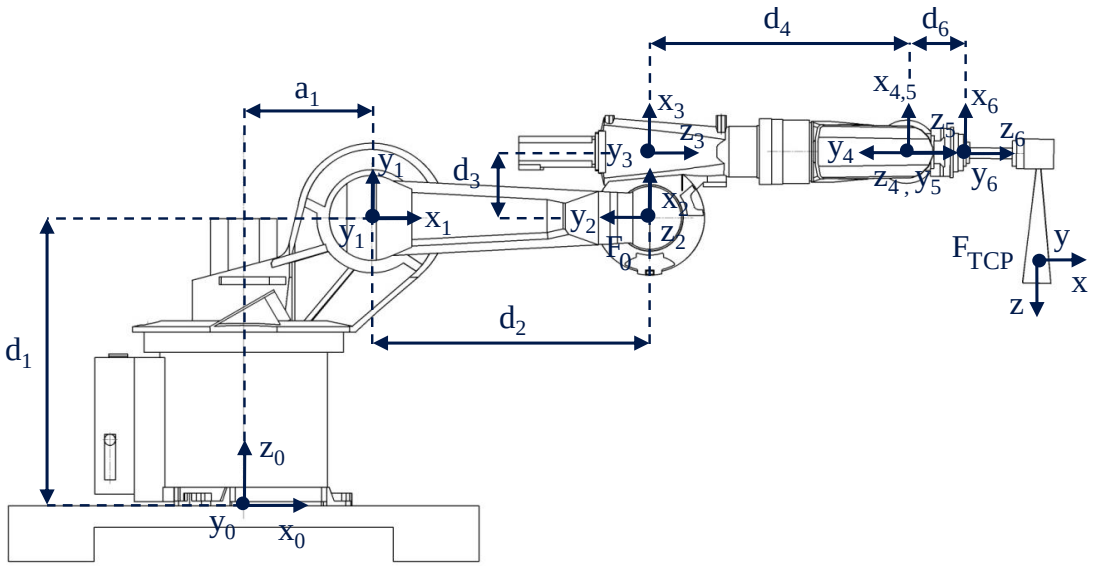


Figure 4.4: Coordinate frame assignment to the FIS.

According to DENAVIT & HARTENBERG (1955), this scheme can be semantically generalized and applied to compute the relationship between arbitrary adjacent frames F_{i-1} and F_i belonging to the robot kinematic chain. Thus the matrix equation (4.3) becomes

$$^{i}T_{i+1} = Rot(\theta_i) \times Trans(d_i) \times Trans(a_i) \times Rot(\alpha_i) \tag{4.4}$$

which can be concatenated to

$$^{i}T_{i+1} = \begin{bmatrix} cos\,\theta_i & -cos\,\alpha_{i+1}\,sin\,\theta_i & sin\,\alpha_i\,sin\,\theta_i & a_i\,cos\,\theta_i \\ sin\,\theta_i & cos\,\alpha_{i+1}\,cos\,\theta_i & -sin\,\alpha_i\,cos\,\theta_i & a_i sin\,\theta_i \\ 0 & sin\,\alpha_i & cos\,\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5}$$

The $^{i}T_{i+1}$ matrix in equations (4.4) and (4.5) is known as the DH transformation matrix and the four parameters $[\alpha_i\ a_i\ d_i\ \theta_i]$ are called the DH parameters (DENAVIT & HARTENBERG 1955). Hence, the FIS chain can be modeled kinematically by the values of these four quantities for each link. α_i and a_i describe the link's connection to the predecessor's neighboring link, θ_i and d_i describe the link itself. Table 4.1 summarizes the DH parameters corresponding to the FIS frame placement as depicted in figure 4.4. Before the frame assignment, the FIS is outstretched in its zero position, where all joint variables θ_i correspond to zero.

Table 4.1: DH-parameter of the FIS

***Link* (*i*)**	$\boldsymbol{\alpha_i}$	$\boldsymbol{a_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$
1	90°	a_1	d_1	θ_1
2	0	a_2	0	θ_2
3	90°	*a3*	0	θ_3
4	-90°	0	d_4	θ_4
5	90°	0	0	θ_5
6	0	0	d_6	θ_6

4.3.2 Kinematic Analysis of the FIS

There are two principal problems in kinematic analysis, one referred to as *forward kinematics*, the other as *inverse kinematics*. The former is concerned with the problem of determining the manipulator TCP pose when its joint angles θ_i are given. The latter refers to the converse problem of finding the magnitude of joint values θ_i, which results in a desired TCP pose. In mathematical terms, this means that solving

the forward kinematic problem is consistent with mapping the robot joint space onto Cartesian space, whereas the inverse kinematic solution maps the other way around.

4.3.2.1 Forward Kinematics

Once the FIS's DH parameters are known, deriving its forward kinematics is straightforward. It is obtained by establishing a relationship between the FIS's base frame $F_0(x_0\ y_0\ z_0)$ and its TCP frame F_{TCP}. Assuming that the homogeneous transforms ${}^6T_{TCP}$—describing the TCP's frame with respect to the robot's flange F_6—is known, the forward kinematics are calculated by consecutively post-multiplying the individual link transformations ${}^iT_{i+1}$ across the manipulator chain as shown in equation (4.6)

$$ {}^0T_{TCP} = \prod_{i=0}^{5} {}^iT_{i+1}(\theta_{i+1}) \times {}^6T_{TCP} \tag{4.6} $$

This yields a single homogeneous transform ${}^0T_{TCP}$ that relates the frame F_{TCP} to the frame F_0 and therefore allows for the calculation of the TCP once the joint angles θ_{i+1} are given.

4.3.2.2 Closed-Form Inverse Kinematic

The goal of the inverse kinematics is to choose joint values that will place the TCP frame F_{TCP} at a desired location. The most straightforward approach for solving the inverse kinematics problem analytically also draws on equation (4.6), but this time ${}^0T_{TCP}$ is specified and can be expressed as

$$ {}^0T_{TCP} = \begin{bmatrix} n_1 & o_1 & a_1 & P_x \\ n_1 & o_2 & a_2 & P_y \\ n_1 & o_3 & a_3 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.7} $$

whereas the joint angles θ_i expressed in the DH matrices ${}^iT_{i+1}(\theta_i)$ are unknown. Hence, equation (4.6) becomes

$$ \prod_{i=0}^{5} {}^iT_{i+1}(\theta_{i+1}) \times {}^6T_{TCP} = \begin{bmatrix} n_1 & O_1 & a_1 & P_x \\ n_2 & O_2 & a_2 & P_y \\ n_2 & O_3 & a_3 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.8} $$

The entries of the left-hand side matrix in equation (4.8) are non-trivial trigonometric functions of the joint angles θ_i. Equating them element-wise to the scalar

entries of the matrix on the right-hand side of equation (4.8) results in twelve nonlinear equations of six unknown variables that are unsolvable in closed-form[8] (SPONG ET AL. 2006 p. 95). Since the manipulator type considered in this study own a spherical wrist, a closed-form solution can be derived (PIEPER & ROTH 1969). Closed-form inverse kinematic solutions bear the advantages of speed and computational stability and, perhaps most meaningfully, they show exactly where the various singularities will occur (LLOYD & HAYWARD 1993).

Borrowing from the findings documented in PIEPER & ROTH (1969), the FIS's inverse kinematic can be solved in two steps. First, the joint angles of the first three manipulator's actuators that position the *wrist center point*[9] (WCP) are determined so that the desired TCP pose is realized. Next, the last three joint angles that appropriately orient the TCP are algebraically calculated. Figure 4.5 illustrates the calculation of the three first joints θ_1, θ_2 and θ_3 with a planar geometric approach using purely trigonometric relationships. This is achievable owing to the orthogonal structure of the manipulator e.g., the angle between neighbor axes are either 0° or multiples of 90°.

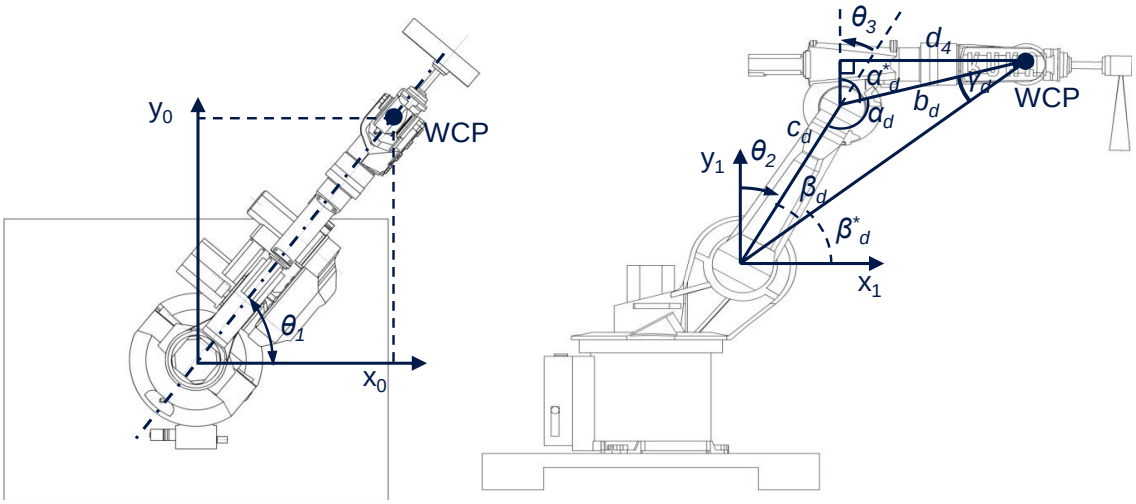


Figure 4.5: Geometrical calculation of θ_1, θ_2 and θ_3

The wrist joint variables θ_4, θ_5 and θ_6 are determined by rewriting equation (4.8) as follows:

$$\prod_{i=0}^{2} {}^{i}T_{i+1}(\theta_i) \times \prod_{i=3}^{5} {}^{i}T_{i+1}(\theta_i) \times {}^{6}T_{TCP} = \begin{bmatrix} n_1 & O_1 & a_1 & P_x \\ n_2 & O_2 & a_2 & P_y \\ n_2 & O_3 & a_3 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.9}$$

[8] Refers to a solution method based on analytic expressions or on the solution of a polynomial of degree 4 or less, such that non-iterative computation steps are sufficient to deduce a solution (PIEPER 1968, P.382).

[9] The wrist center point is the point where the last three revolute joint axes of the manipulator intersect.

The equation to be solved for the final three joint variables is therefore

$$\prod_{i=3}^{5} {}^{i}T_{i+1}(\theta_i) = \prod_{i=0}^{2} ({}^{i}T_{i+1}(\theta_i))^{-1} \times \begin{bmatrix} n_1 & O_1 & a_1 & P_x \\ n_2 & O_2 & a_2 & P_y \\ n_2 & O_3 & a_3 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times ({}^{6}T_{TCP})^{-1} \quad (4.10)$$

since ${}^{i}T_{i+1}(\theta_i)$ can be calculated once the first three joint angles are known, the matrix on the right-hand side of the equal sign consists of known entities. The left-hand side of equation (4.10) contains trivial trigonometric functions of the joint angle values to be found. By equating the two matrices element-wise, it is now a simple matter to calculate θ_4, θ_5 and θ_6. As the link lengths d_3 and d_5 are zero, the inverse solution is not unique. Each of the joint-variables θ_1, θ_2 and θ_5 has two solutions. Thus, a maximum of eight possible combinations of θ_i for a simple pose of the robot can be derived. Though only one of the eight multiple solutions is sufficient to perform the position and orientation of the TCP, all the solutions are computed. An algorithm which will be introduced in chapter 7 will then decide which one is to be chosen as the "optimal" solution with respect to various optimazation criteria.

It should be noted that, when the pose to be reached by the TCP is out of the reachable robot workspace, the joint values obtained from the inverse kinematic will have imaginary values, indicating that no practicable solution exists. Moreover, not all real solutions will always correspond to a physically realizable configuration of the FIS. For instance, the motion of the revolute joint may be restricted by rigid limit stops. Since the formulation of the inverse kinematic does not include constraints on the joints limits, these are accounted for by testing the solution computed against the upper and lower travel range bounds of each joint.

4.4 Collision Engine

When operating robotic systems, it is of central importance to avoid interference between the robot and its surroundings. Hence a collision observer, capable of detecting impending collisions, was plugged into the simulation module. Numerous methods and algorithms have been devised by researchers to tackle the issue of collision checking between meshed CADs. From all these, the Proximity Query Package (PQP) developed by LIN & MANOCHA (2004) that is purported to be computationally lesser "greedy" has been fitted in the simulation module. Another reason for choosing PQP was its ease of use and, in particular, the fact that a clear interface description is provided and almost no parameter tuning is required.

The simulation module uses the PQP as a black-box collision engine to detect interference between the virtual FIS and other workcell items. For this purpose, the PQP was given access to the geometrical model of the FIS workcell. At each simulation step, the PQP is queried and its responses (e.g. colliding items, intersecting triangles, penetration depth of the intersecting triangles) are interpreted. Any pending collision is instantly sent back to the caller routine and displayed within the visualization window. Figure 4.6 depicts a collision situation where the *link5* of the FIS interferes with the WP. The offending parts are darkened to draw the attention of the operator.

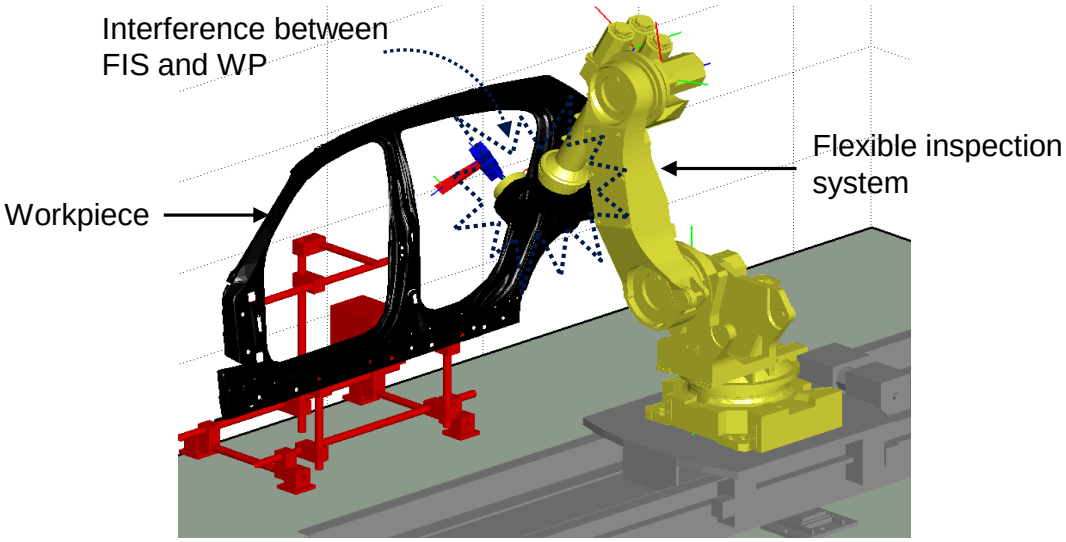


Figure 4.6: A colliding configuration of the FIS

4.5 Virtual Sensor

The virtual sensor should be understood as a software module that emulates the real LSS. It will be used as a validation tool to predict the scan results of pre-computed scan trajectories before they are downloaded to the real FIS. The virtual sensor is based on a sweep-plane and a ray tracing approach and may be logically structured into: *a mathematical model, a laser source model,* and *a camera model*.

4.5.1 Mathematical Model

An LSS can be mathematically described by specifying its geometrical characteristics and viewing parameters. As illustrated in figure 4.7, the geometrical characteristics of a LSS encompass:

- **The laser plane:** a triangular-fashioned plane that contains the laser beam;

- **The sensor axis**: the laser plane symmetry axis;
- **The triangulation angle:** the angle between the laser plane and the camera axis;
- **The aperture angle:** dihedral angle of the emitted laser plane;
- **The focal point:** point located at the sensor's focal distance, e.g. where the camera focuses most sharply;
- **The camera view frustum:** pyramid representing the portion of the view space that is projected onto the photosensitive detector of the camera;
- **The depth-of-field (DoFi):** range of distance an object has to lie in to stay in focus of the camera;
- **The field-of-view (FoV):** clipped area of the laser plane delimited by the DoFi and the LaWi. It represents the measuring area of the sensor; i.e. only objects that lie in the FoV can be imaged;
- **The laser width (LaWi):** the spread of the laser plane at the end of the FoV. This does not really represent an independent parameter since it can easily be derived by considering trigonometric relationships between the aperture angle and the DoFi.

The viewing parameters, in contrast, consist of the position and orientation of the sensor frame F_{FP}. The latter is rigidly attached to the focal point that happens to be the FIS's TCP. In keeping with standard practice in robotics, the unit vectors of F_{FP} bear the labels $[\vec{s}\ \vec{d}\ \vec{a}]$, which stand for slide, direction and approach vector. The approach vector is collinear to the scan axis pointing away from the laser source. The slide vector is orthogonal to the normal vector and aligned with the laser plane. Finally, the direction vector is orthogonal to the plane approach and sliding vector to complete the right-handed orthogonal coordinate frame F_{FP}. Further relevant frames bounded to the sensor are: the camera frame F_{CA}, the laser source frame F_{LS} and the frame F_{RI} attached to the sensor mechanical interface (see figure 4.7).

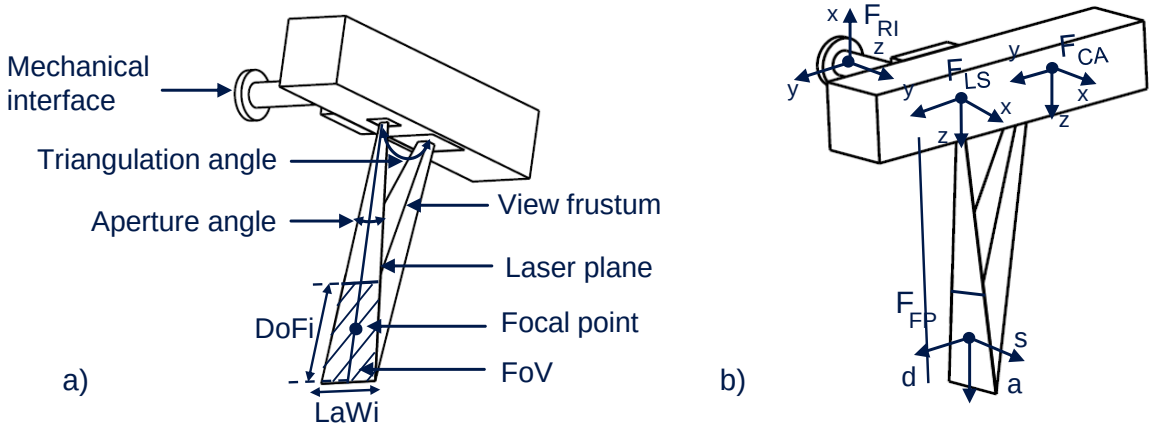


Figure 4.7: Mathematical model of the sensor: a) Geometrical characteristics; b) Sensor frames

4.5.2 The Laser Source Model

4.5.2.1 Ray-triangle Intersection

The laser source model is based on ray tracing, which is a rendering technique commonly used in computer graphics to simulate light rays in a virtual scene. For more general references on the topic of ray tracing, the readers are directed to MÖLLER ET AL. (2008). As the ray triangle intersection constitutes the tenet algorithm of this technique, a simple implementation of it is briefly introduced here. Figure 4.8 graphically illustrates the geometrical interpretation of the barycentric ray-triangle intersection problem.

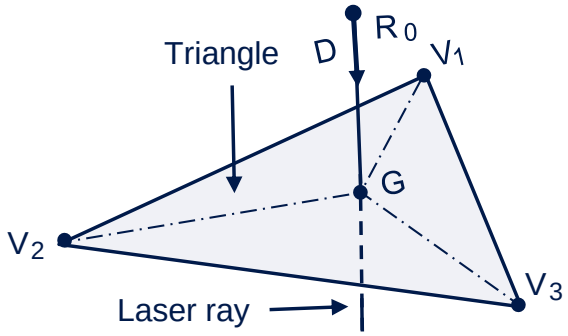


Figure 4.8: Illustration of the barycentric ray triangle intersection

Considering the parametric equation of a line with origin R_0, normalized direction $\vec{D}$ and a real parameter t

$$R(t) = R_0 + \vec{D}.t \tag{4.11}$$

and that of a plane containing a triangle (V_1, V_2, V_3),

$$P(u,v) = (1-u-v)V_0 + uV_1 + vV_2 \tag{4.12}$$

where $(u, v \; and \; 1-u-v)$ are the barycentric coordinates obtained from the ratios of the sub triangles (V_1, V_2, G), (V_1, V_3, G) and (V_2, V_3, G).

The intersection of both can be found by equating (4.11) and (4.12). This new equation, along with (4.11) and (4.12), yields an equation system which allows the identification of the three unknowns: u, v and t. As the solution of this equation system is trivial, it is omitted for brevity. If there is a real solution for this equation system, then the constraints in equation (4.13) are used to determine whether the solution lies inside the triangle (thus an intersection) or outside the triangle (no intersection).

$$(u \geq 0, v \geq 0, u+v \leq 1) \tag{4.13}$$

4.5.2.2 Laser Source Model

The laser source is conceptualized as a ray emitting source, which radiates toward the focal point (see figure 4.9). The rays are constrained to the laser plane and their individual directions are angularly uniform across it. The number of rays (n_{LS})is tunable and, along with the aperture angle α_{LS}, determines the sensor resolution. Whereas α_{LS} is limited to maximum 60°, no restriction is imposed in n_{LS}.

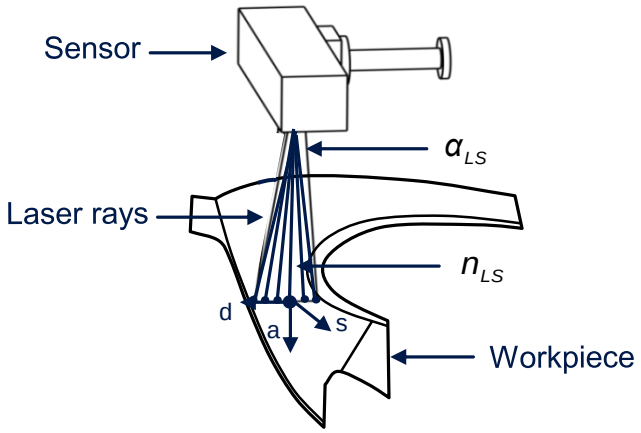


Figure 4.9: Laser source model

The rays are constrained only to be able to interact with triangles of the WP. Since the ray triangle intersection algorithms are computationally expensive (PURCELL 2004 p. 10), special care is taken to reduce their computing cost. Capitalizing on the parallelization capabilities of modern CPUs, the contribution of each ray to the final image is computed independently from the other rays within a thread. The threads are then dispatched to the different cores of the CPU to achieve efficient load balancing.

During a virtual scan process, the laser plane is swept across the meshed model of the WP. The ray radiation flux coming down from the laser source is shone on a collection of WP's facets and each of the n_{LS} rays is tested for intersection with the WP. The impact distance of the intersecting ray is computed by applying the ray casting algorithm introduced above. Rays that miss the object's surface are ignored. To mimic the DoFi, the ray-tracing algorithm is adapted to consider only intersection points which lie in a given distance interval from the laser source. For every scan line slice through the mesh, the resulting data is returned as an $n_{LS} \times 3$ array of intersecting points. They are then expressed in the WP frame but may also be transformed to any other frames. During a continuous measuring sweep, the data of each scanned line are held in a record array, which is continuously updated with new scanned data lines. At the end of the digitization process, the simulation results are conveyed to the graphical interface for subsequent rendering on the GUI.

4.5.3 The Camera Model

In its current state, the laser source model would test against several non-involved WP facets before finding the right ones. Using such a brute force approach is an option only if the number of facets is negligible, otherwise this will considerably slow down the scan simulation. To remedy this drawback, facets which cannot be intersected by the rays have to be culled away, as suggested in MÖLLER ET AL. (2008 pp. 646–713). This is where the camera model comes into play. The main purpose of the camera model is to apply object-partitioning techniques on the WP model to reduce the amount of ray-triangle intersection tests at each simulation step. Consequently, two partitioning techniques common in the field of graphic simulation have been implemented in this model.

The first object partitioning technique is *backface culling*. The idea behind this is to eliminate polygons facing away from the sensing direction, since triangles that do not contribute to the range image do not need to be considered. Implementing this algorithm for a facetted model is straightforward, since STL facets have an orientation (see section 4.2.1). The core of the implemented *backface culling* algorithm computes the dot product of the sensor's approach vector $\vec{a}$ and the triangles' normal vectors. A positive dot product means that the angle between the two vectors is less than $\pi/2$ radians, so the triangle is not front-facing the sensor and can be safely ignored by the ray-tracing algorithm.

While *backface culling* will remove about half of the polygons in the meshed model of volumetric objects, it provides little gain in the case of thin-walled objects such as sheet metal parts. For instance, the part depicted in figure 4.10 has most of its surface facing the sensor system, so there are relatively few facets to cull away. Consequently, for the same purpose of decreasing the computational load devoted to the sensor model, a second object partitioning algorithm called *frustum culling* (MÖLLER ET AL. 2008 pp. 660–665) is applied downstream to the *backface culling*. It operates by figuring out which facets totally or partially fall into the camera frustum at a specific sensor viewpoint. In doing so, a subset of triangles that do not include those triangles that are too far away from the camera view frustum is extracted from the densely meshed models. On account of this, at each simulation step, right before the ray-tracing algorithm engages, the camera frustum is tested for collision against the WP's model using the collision engine PQP. The result of the collision query is then interpreted as the subset of facets that can be intersected by rays launched from the laser source (see frame a) on figure 4.10). Non-colliding facets are ignored for the simulation step during which they were singled out as "non-colliding." To counteract any adverse effects that may be provoked by tolerance and numerical imprecision, the frustum CAD model is slightly dilated. An

isotopic dilatation coefficient of 1.05 proved to yield satisfying results.

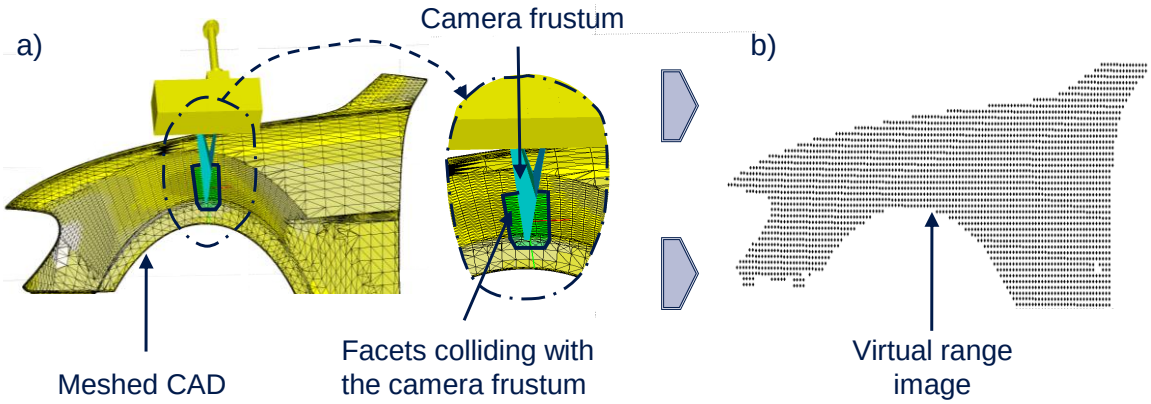


Figure 4.10: a) Frustum culling (the highlighted facets are those which currently interfere wth the dilated camera frustum); b) Simulated range image

Frame b) of figure 4.10 depicts the results of a simulation run carried out to access the qualitative performance of the virtual sensor. This picture shows a simulated range image resulting from the complete virtual digitization of the sheet metal WP whose CAD is depicted in the frame a) of the same figure. As this range image matches closely with the geometry of the CAD model, one can deduce that the virtual sensor satisfies the requirements for which it was modeled. It is important to note, that this virtual range image differs slightly from the real one that would have been obtained with a physical sensor. A reason for this is that physical effects such as multiple reflections, diffractions, noise and color sensitivity have not been considered in the virtual sensor model. However, this image is sufficient to get a first feeling for the efficiency of generated scan trajectories prior to delivering them to the physical FIS for execution.

5 Task Description Module

5.1 Overview and Module Structure

A major tenet of ToP systems is their ability to specify programming tasks at a high-level of abstraction compared with that of regular robot programming languages. Accordingly, this module will introduce a task description method based on feature contour lines. The method is geared especially towards operators who may not have been intensively trained to deal with robotic and metrology equipment. Prior to addressing this issue, attention is devoted to the problem of adequately positioning WPs within the FIS's workspace. This—at first glance trivial-sounding—problem may actually handicap a programming task in the sense that inappropriate WP location may cause automatic motion planning algorithms to fail. Figure 5.1 depicts the overview and features to be implemented in this module.

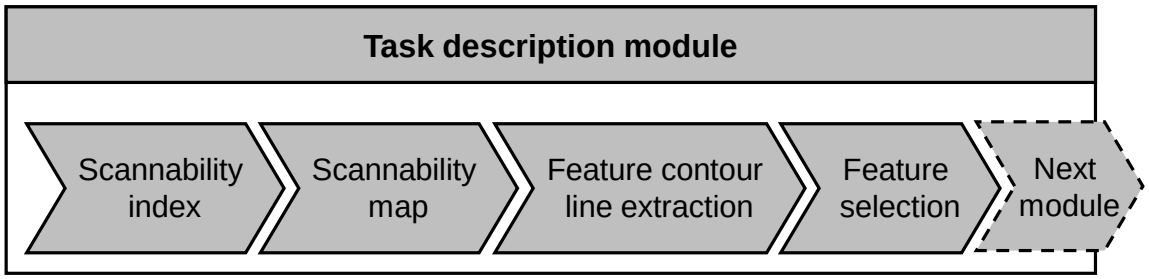


Figure 5.1: Module features

5.2 Solving the WP Positioning Problem

5.2.1 Preliminaries

Obviously, for the FIS to perform a digitization task on a WP, the latter should be reachable by the sensor's FoV. Robot manufacturers usually provide a working envelope for their robots, which is nothing more than the space reachable by the robot's flange (see figure 5.2). However, for manipulators equipped with a non-trivial end-effector, such an envelope offers little assistance except for rough evaluation purposes. Another drawback of such working envelopes is that they do not capture any information about the manipulator's *dexterous workspace*, i.e. the subspace of a manipulator workspace that can be accessed with arbitrary end-effector orientations (CRAIG 2005 p. 102). Most regions of the manipulator-reachable space are however not accessible with random orientation due to some physical limitations inherent to the manipulator's mechanical structure. Thus,

special care has to be taken at the early stages of robot workcell planning to prevent invalid or suboptimal workcell layout (LUETH 1993 p. 2–3, MITSI ET AL. 2008, BERBYUK 2010).

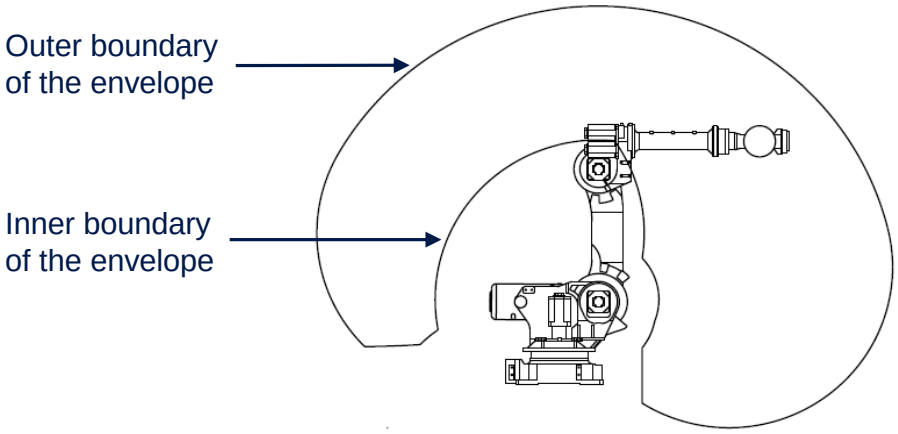


Figure 5.2: Spherical working envelope of a typical manipulator

Despite previous efforts to automate robotic workcell layout design, workcell layout design is still intuitively determined by highly skilled designers (KONIETSCHKE 2007 p. 32). Relying on their experience, they proceed by trial and error as well as interactive checks in order to appropriately arrange the workcell (BARRAL ET AL. 2001). Such a process is time-consuming and therefore not practical in automatic robot programming. In the following sections, a numerical solution method is laid out which can spot the workcell areas where the WP should be placed in order to guarantee maximum accessibility for the FIS.

5.2.2 The Scannability Metric

For the purpose of capturing the FIS's scanning ability within its workcell, the theoretical quantity *scannability index* is introduced. This is a robot-dependent metric that captures the ease of arbitrarily locating the FIS's TCP at a specific point. For a given location P within the FIS's reachable space, a *scannability index* is computed according to the following three steps.

Step 1: Orientation discretization

First of all, the space surrounding P is sampled to generate several sensor orientations from which P can be captured. To this end, an anchor frame I_0 whose axes are parallel to the reference frame F_{WC} is assigned to P. Further frames I_i are cloned from frame I_0 by applying a discretization scheme characterized by three parameters: azimuth angle φ_{az}, elevation angle φ_{el}, and a tilt angle φ_{ti}, respectively. All of these are expressed relative to the axes of the anchor frame I_0. The frames I_i are spawned by incrementing the three parameters mentioned above in a range from 0°

to 360°. The incrementation step $\Delta\varphi$ can be freely chosen. Nevertheless, to ensure maximal coverage, an upper bound equal to the sensor aperture angle is imposed on $\Delta\varphi$. Since computational load is expected to grow exponentially with $\Delta\varphi$, it dictates a lower bound on $\Delta\varphi$. Figure 5.3 illustrates the meaning of these three discretization parameters geometrically.

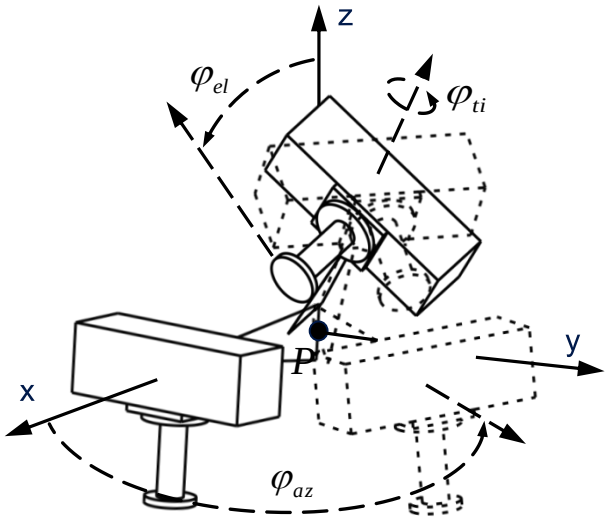


Figure 5.3: Orientation discretization scheme

Step 2: Investigating the ease of capturing the point P

For every frame I_i, an attempt is made to solve the inverse kinematic for all possible solutions. Sometimes, the inverse kinematic may output fewer than the eight solutions, indicating restricted access of the FIS to the frame I_i. The missing solution may be caused by joint values that lie outside the joint travel ranges or be purely nonexistent if the frame I_i engenders a singular configuration of the FIS. To escape such handicaps, the procedure explores the close vicinity of P for points with a higher number of valid inverse kinematic solutions. The idea behind this strategy is to take advantage of the sensor's ability to collect any point located within its FoV. Hence, even though a specific target frame I_i maybe not reachable by the FIS's TCP, this doesn't necessarily mean that P_k is not "scannable" from this orientation. Indeed, to scan a point, it is enough to have it within the sensor's FoV and not necessarily at the sensor focus point or TCP.

Having said that, each time the inverse kinematic does not return eight solutions, an additional six satellite points $P_{1...6}$ are spawned. These points are arranged equidistantly on a circle concentric to P, as shown in figure 5.4 (for the sake of clarity only four satellite points are depicted). Next, the FIS's TCP is driven to each satellite point and the inverse kinematic is queried again. If more solutions are returned at a satellite point than originally at P, then these new solutions are substituted to those

previously obtained at P. In this way, P will be considered to have better *scannability* attributes, even though it is itself not reachable by the FIS TCP. As this step is carried out, a record is kept about the theoretically possible inverse solutions N_{th} and actually valid solutions N_{va} at each I_i.

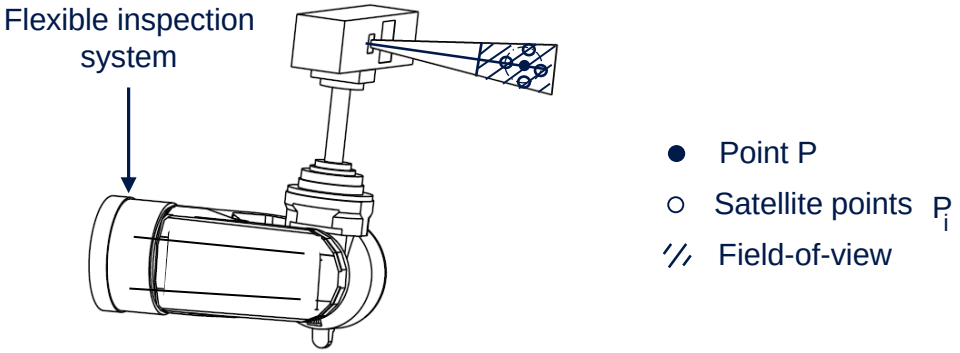


Figure 5.4: Escaping invalid inverse kinematic solutions caused by singularities or axis limits

Step 3: Scannability index

Once step two is completed, a numerical measure, termed the *scannability index* S_i, is derived by setting N_{th} and N_{va} into proportion as follows:

$$S_i = \frac{N_{va}}{N_{th}} \tag{5.1}$$

S_i represents a quantitative performance index, which captures the FIS's scanning ability at a given point. As the condition $N_{th} \geq N_{va}$ always holds, so does $S_i \in [0\ 1]$. Low or zero S_i values indicate poor or nonexistent scanning abilities of the FIS at the corresponding point, whereas higher S_i values predict better scanning aptitude.

5.2.3 The Scannability Map

To capture the FIS's *scannability* not only at a point but also rather in the entire workcell, the procedure is repeated across the latter. The first step in doing so consists of the sampling of the working space surrounding the FIS robot into a 3D grid of evenly spaced nodes, as schematically depicted in figure 5.5. It goes without saying that the accuracy of the *scannability* analysis heavily depends on the sampling step. It is also true that what holds for the orientation discretization parameter $\Delta\varphi$ also holds for the grid sampling parameter Δd, namely that the computation time exponentially increases with their resolution. To provide high resolution without excessive computational requirements, only a subspace of the workcell may be taken into consideration.

For each grid node, a s*cannability index* is derived by the procedure designed above. The results are stored in a three dimensional data structure whose topology resembles that of the grid. After the computation stage, the data structure is visualized by a 3D colored map. This map—the *scannability map*—encodes an abstract representation of the FIS's capabilities and, in this way, it simultaneously characterizes which portion of the workcell can be easily captured by the FIS. Since the *scannability map* is computed with respect to the manipulator base, the *scannability analysis* has to be carried out only once for each FIS. If the FIS is replaced within the workcell, the map only needs to be shifted accordingly.

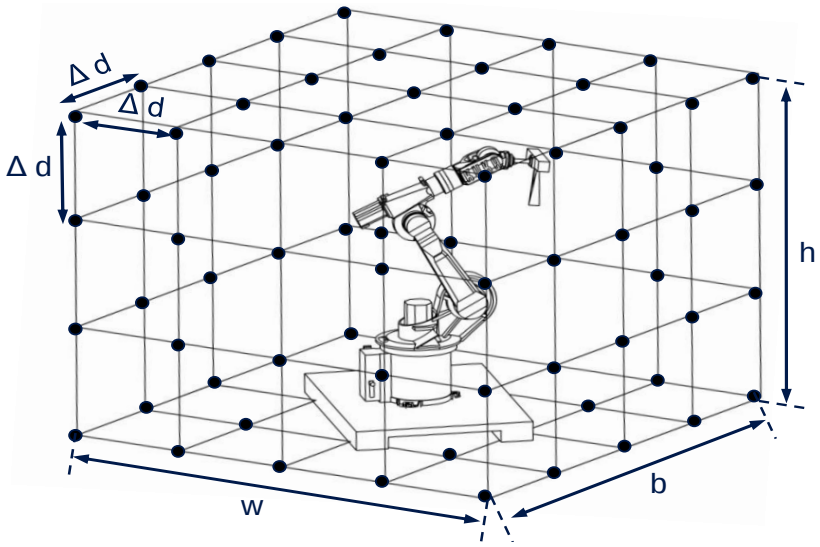


Figure 5.5: Workcell discretization scheme

5.2.4 Implementation Details and Simulation Study

Since the *scannability index* at each node can be independently calculated, the computation load is broken down into multiple threads that run simultaneously. The maximum number of threads is limited by the number of computing cores available on the computing platform. The numerical values of the sampling parameters adopted during the experiment are $\Delta\varphi = 15°$ for the orientation discretization and $\Delta d = 20\ mm$ for the workcell sampling. The computation time on the development platform[10] amounted to approximately 32 hours. However, there is no cause for concern as the *scannability map* is intended to be performed in a pre-processing step preceding the programming stage.

Figure 5.6 visualizes a grayscale image of the *scannability map*. The darkest colored areas have poor *scannability attributes* and represent the portion of the workcell where the WP should not be positioned. Gray marked regions have

[10] Windows XP-based workstation equipped with 2.7 GHz 4 cores Intel processor and 3GB RAM.

medium *scannability attributes* and are likely to provide mediocre FIS performance in the case of large or complex-shaped WPs. Bright areas denote those areas that exhibit the greatest *scannability attributes*. Positioning a WP there will guarantee the easiest access by the FIS.

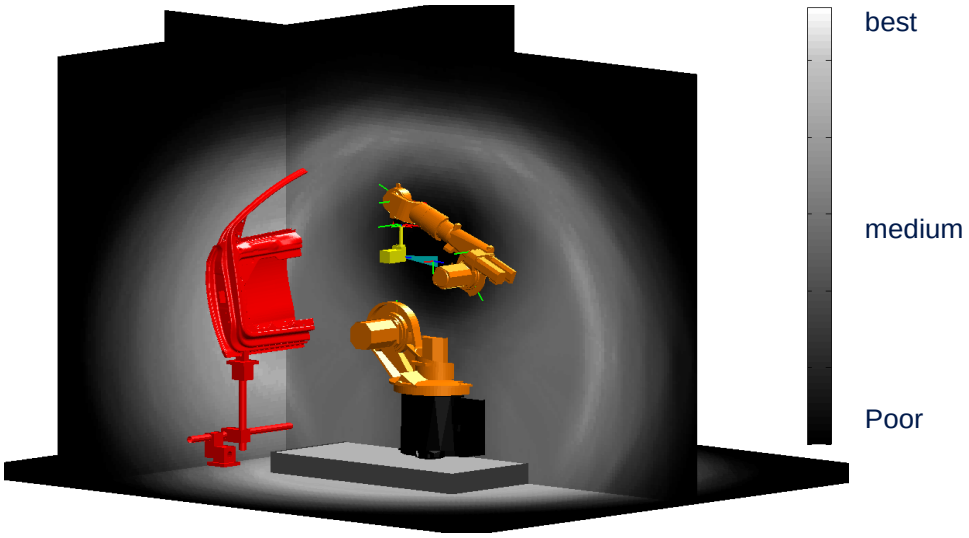


Figure 5.6: Scannability map of an FIS. The map is sliced by two vertical planes for better visualization.

5.2.5 Discussion

In the light of the above results, it can be affirmed that an answer has been proposed as to where to place a WP within a FIS workcell. Resorting to heuristic algorithms, it could have even been possible to go a step further and compute the optimal position and orientation of a given WP within the workcell. But since every part would result in a different position, this would have implied the need of flexible fixturing devices such as the one proposed in JONSSON & OSSBAHR (2010). As such devices are costly and highly complex, they are not economically affordable for SMEs. Thus it is deemed best to instead assist workcell designers with visualization tools such as the *scannability map* and leave them the flexibility to choose a convenient WP location, relying on their cognitive capabilities.

5.3 Feature Extraction

5.3.1 Taxonomy of Features

Since the WP design features are key to task description method embodied in this module, a brief definition of the term "feature" is given in this section. In fact, the

notion of “feature” is ambiguously coined in the realm of computer-aided design. For instance, WIERDA (1990) defined a feature as *"a partial form or a product characteristic that is considered as a unit and that has a semantic meaning in design, process planning, manufacture, cost estimation or other engineering discipline."* In contrast, EHRLENSPIEL ET AL. (2007 p. 489) assert that features are artifacts for the description of WPs that represent functional, geometric and technological properties. However verbalized, all terminologies related to the word "feature" seem to share the idea that features play the central role in the design of products (WOOD 1994 p. 26). According to TECH ET AL. (2010), features can be classified as form, inspection (tolerance), material, functional, manufacturing and mating (assembly) features, depending on their intended use. With reference to the scope of this work, only form features, i.e. features that serve for the geometric description of a product, are of interest. Therefore, for the purpose of this research, just as in WANG & OZSOY (1991) and in SALOMONS ET AL. (1993), features are equated with specific artifacts on the surface WPs that modify their appearance, including edges, corners holes, slots, pockets and chambers.

In several manufacturing-related engineering disciplines it is required that form features are automatically extracted from a product CAD model in order to automate particular planning processes (VANDENBRANDE & REQUICHA 1993, ELMARAGHY ET AL. 2009). Depending on the standard used to store the product model, this task may be more or less trivial in term of implementation effort. For instance, in the IGES[11] and STEP[12] data format, features, as well as volumetric and surface information, are explicitly stored in the data structure. In such a case, retrieving the feature data is achieved by simply indexing them in the file data structure (BHANDARKAR & NAGI 2000). For CAD data formats that do not explicitly record feature data, extracting them is no longer a trivial task. Examples for this are triangularly meshed product models that have not been originated from another solid model but from a point cloud or a range image. This is the case for prototype parts that are manually shaped in malleable materials (clay, synthetic foam, plaster), then completely digitized and subsequently reconverted into meshed CAD model (CURLESS 1999, MAHR 2003, SONG & SHIMADA 2009). Such meshed models only capture the boundary representation (B-rep) of the product surface. Neither semantic nor topological information regarding the part features is explicitly stored in the data format structure (ROCK & WOZNY 1992). As a consequence, features can only be extracted by applying analytical or discrete calculations methods on the mesh (ROESSL 2005).

[11] IGES stands for Initial Graphic Exchange Specification and represents the first attempt (1979) to create a vendor-neutral format that enables a seamless interchange of CAD models across different CAD systems.

[12] STEP stands for STandard for the Exchange of Product Model data and represents a neutral formal standard created to foster information sharing across various CAx applications (ISO 10303).

5.3.2 Feature Contour Extraction from Meshed CAD Models

5.3.2.1 Preliminaries

For the sake of generality, the feature extraction method to be elaborated in the following sub-sections will exclusively focus on the non-trivial case of meshed CAD models. At this juncture it is imperative to clarify that the feature extraction method to be introduced below is not aimed at identifying or recognizing the topology of the feature but rather at isolating their contour lines, as this is enough for the description of the digitization task. The feature extraction method is based upon techniques from the field of differential geometry and discrete image processsing introduced in PETITJEAN (2002), PAULY ET AL. (2003), and GATZKE ET AL. (2006) and operates in five stages: *a pre-processing*, *a classification*, *a selection*, *a skeletonizing* and finally, *a pruning stage*. Before expounding on these stages, a closer look will be taken into how to estimate the surface curvature on meshed surface models, as this constitutes the pivotal idea of the feature extraction method.

5.3.2.2 Curvature Estimation on Meshes

Surface curvature, henceforth referred to as curvature, can be understood as a metric intrinsically describing how a surface behaves in a local region (MEYER ET AL. 2002). In CAD models, design features distinguish themselves by high curvature magnitude (ROESSL 2005). From this it follows that features can be extracted by tracking surface curvature changes in the CAD model. In order to analytically determine the curvature of a free-form surface, it must be smooth and at least two times differentiable (PRESSLEY 2009 p. 31). However, a meshed surface represents only a first-order approximation of smooth surfaces it has been generated from. Hence, the surface curvature in meshes CAD can only be numerically approximated (GATZKE ET AL. 2006). Based on the natural parameterized two-times differentiable curve f depicted in figure 5.7, a simple discrete curvature approximation will be derived.

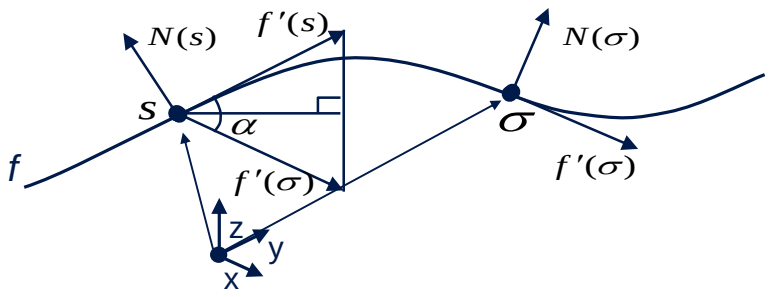


Figure 5.7: Parametric plane curve f with unit tangent vectors $\vec{f'}(s)$ and $\vec{f'}(\sigma)$, normal vectors $\vec{N}(s)$ and $\vec{N}(\sigma)$ at two cuve point s and σ

From differential geometry it is known that the normal curvature $k(s)$ at any curve point s of f can be expressed in terms of the second derivative of f at this point:

$$k(s) := |f''(s)| = \left|\frac{d^2 f(s)}{d^2 s}\right| = \frac{d}{ds}\left(\frac{df(s)}{ds}\right) = \lim_{s\to\sigma} \frac{f'(\sigma) - f'(s)}{\sigma - s} \tag{5.2}$$

Considering two closely spaced points s and σ on f, the discrete setting of equation (5.2) translates to

$$k(\sigma) \approx \frac{|f'(s) - f'(\sigma)|}{|s - \sigma|}, \qquad for \;\; |s - \sigma| \approx 0 \tag{5.3}$$

From Figure 5.7 it can be deduced that

$$\frac{|f'(s) - f'(\sigma)|}{|s - \sigma|} = \frac{2\sin\frac{\alpha}{2}}{|s - \sigma|} \approx \frac{\alpha}{|s - \sigma|}, \;\; since \;\; \alpha \ll 1 \tag{5.4}$$

where α denotes the dihedral angle between $\vec{f'}(\sigma)$ and $\vec{f'}(s)$. Subsequently, substituting equation (5.4) in (5.3) leads to

$$k(s) \cong \frac{\alpha}{|s - \sigma|} \tag{5.5}$$

As the normal vectors $\vec{N}(\sigma)$ and $\vec{N}(s)$ are orthogonal to $\vec{f'}(\sigma)$ and $\vec{f'}(s)$ respectively, α also represents the dihedral angle between $\vec{N}(\sigma)$ and $\vec{N}(s)$. It can therefore be stated that the normal curvature between two points of a smooth curve can be approximated by applying a second-order difference operator (SOD) that is proportional to the dihedral angle between the normal vectors at those points. Although a discretization error is introduced by the SOD operator if the mesh is densely distributed, this error can be expected to be negligible.

5.3.2.3 Pre-processing Stage

The pre-processor stage is primarily aimed at eliminating topological flaws in tessellated CAD models as these may disrupt algorithms processing the mesh. The core of this stage consists of a consistency check routine that tracks semantic inconsistencies through the mesh. Detectable flaws include redundant vertices, degenerated triangles, unconnected facets and unclosed surfaces. Additionally, facet orientation errors are also trapped. Facet orientation errors become apparent when the normal vectors of two neighboring facets exhibit opposite directions. Meshes that pass the consistency check are stored in a complex data structure that allows for a fast data traversal in order to alleviate the upcoming processing steps.

This data structures keep track of the following:

- the connectivity information in the mesh on a per-edge basis;
- the neighboring triangles of each edge;
- the edges delimiting each triangles;
- the edges sharing a common vertex.

Figure 5.8 shows the meshed CAD model of a sheet metal part that has successfully passed the pre-processing stage. This model will be used to visualize step-by-step how the contour lines are filtered as the model progresses through the upcoming stages. The mesh consists of 84306 vertices, 112406 edges, as well as 28102 facets and normal vectors.

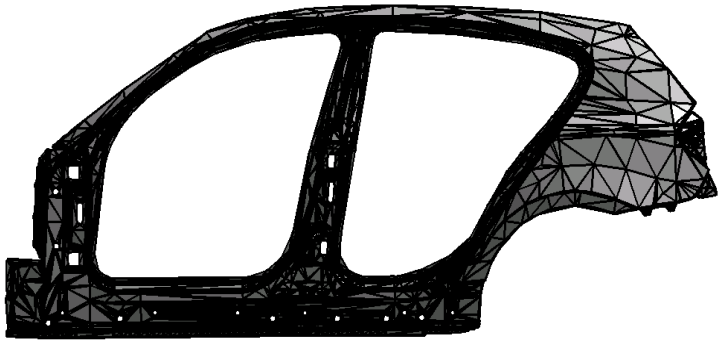


Figure 5.8: Meshed CAD model of a sheet metal part

5.3.2.4 Classification Stage

The goal of this stage is to characterize the edges contained in a CAD mesh using a measure that indicates their likelihood of belonging to a feature contour line. In evidence of the reasons mentioned in section 5.3.2.2, surface curvature lends itself to this endeavor. However, since mesh edges, unlike facets, do not have any normal vector associated with them, the SOD operator cannot be directly applied on them. Beforehand, the normal vectors should be estimated on a per-edge basis. This is accomplished by interpolating between the normal vectors of the incident triangles that share an edge, as shown in figure 5.9. The resulting edge's normal is the average of the neighboring facets' normal biased by their respective distance to the edge.

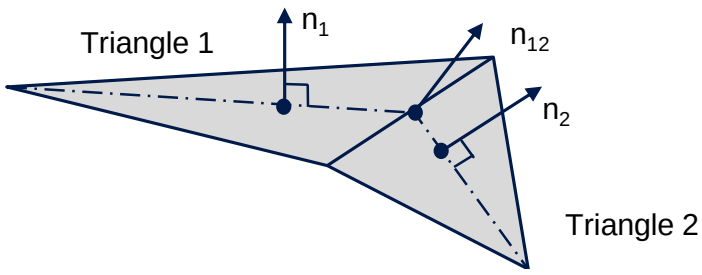


Figure 5.9: Interpolation of the normal of an edge shared by two facets

For a pair of neighboring edges e_i and e_j, associated with the normal vectors $\vec{N}_{e_i}$ and $\vec{N}_{e_j}$ respectively, an approximation of the curvature between them can be found using the equation (5.5). Since the dihedral angle α between $\vec{N}_{e_i}$ and $\vec{N}_{e_j}$ can also be expressed by

$$\alpha = \cos^{-1}\left(\frac{\vec{N}_{e_j}}{\left|\vec{N}_{e_j}\right|} \cdot \frac{\vec{N}_{e_i}}{\left|\vec{N}_{e_i}\right|}\right) \tag{5.6}$$

equation (5.5) can be rewritten as

$$k_{e_1 e_2} \cong \frac{\cos^{-1}\left(\frac{\vec{N}_{e_j}}{\left|\vec{N}_{e_j}\right|} \cdot \frac{\vec{N}_{e_i}}{\left|\vec{N}_{e_i}\right|}\right)}{\left|P_{N_j} - P_{N_i}\right|} \tag{5.7}$$

where P_{N_j} and P_{N_i} designate the location of the involved normal vectors.

5.3.2.5 Selection Stage

This stage is aimed at extracting all edges that may belong to a feature line out of a CAD mesh. A trivial approach to achieving this would have consisted of decimating the mesh by means of a single threshold value that would filter out edges whose curvature factor is lower than the threshold. However, as argued by CANNY (1986), such simple thresholding edge-detection methods are susceptible to streaking[13]. He recommended the uses of more stable *hysteresis filters* for edge-detection tasks in computer vision. In the same spirit, the meshed model is subjected to a *sharp hysteresis filter* to avoid the fragmentation of feature lines. As depicted in figure 5.10, the *hysteresis filter* is characterized by two threshold values: the lower k_{min} and upper limits k_{max} respectively.

Mesh edges whose curvature factor is larger than the upper threshold limit are immediately retained, as they certainly belong to a feature contour line. Those edges will henceforth be labeled as "sharp edges." Mesh edges whose curvature factor is below the lower threshold limit are discarded right away because they are unlikely to be part of any feature line. Edges whose curvature factors are in-between are either retained or discarded depending on the density of "sharp edges" in their geodesic neighborhood. If retained, those edges are also considered "sharp edges"

[13] Streaking is defined as the breaking up of feature contours caused by the threshold filter output fluctuating above and below the threshold value along the length of a contour line (CANNY 1986).

since they are also likely to belong to a feature line.

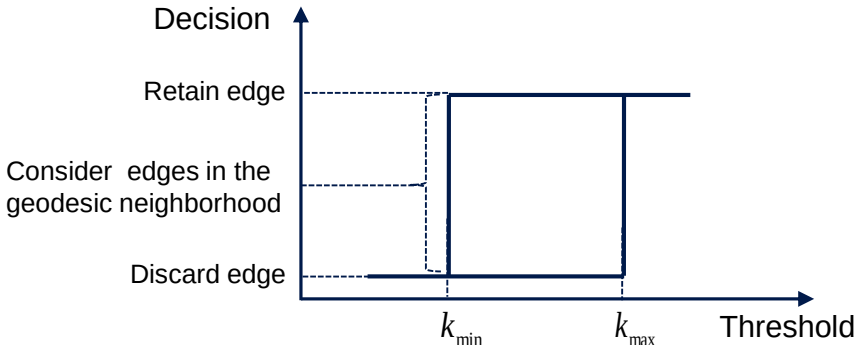


Figure 5.10: Sharp hysteresis filter

The challenge now lies in the proper choice of the two threshold values and the magnitude of the density of sharp edges that qualifies certain edges to be sharp. A sensitivity analysis performed on different meshed models indicated that a threshold lower-upper-limit ratio of two to five within the border of the lowest and highest curvature factor occurring in the mesh leads to stable results. Streaking phenomena were almost nonexistent because for a contour line to be interrupted, the curvature values of its edges must fluctuate above and below the upper and lower threshold, which is unrealistic in view of the gap between these both values. As for the "sharp edge" density, a magnitude of two neighboring "sharp edges" has turned out to be a stable reference value able to suppress outlier edges, which are common in meshed models due to tessellation noise. Once the *hysteresis filter* has acted on the mesh edges, the CAD model is eroded and only those edge patches with edges likely to belong to a feature line remain. Figure 5.11 shows the effect of applying the *hysteresis filter* on the CAD model introduced in figure 5.8.

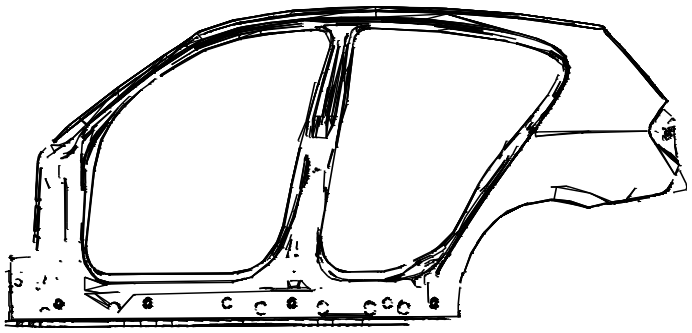


Figure 5.11: Edge patches

5.3.2.6 Skeletonizing Stage

Although the hysteresis filter can be credited with good clustering capabilities, the patches are still too coarse to be considered feature lines. Subsequently, they are

subjected to a skeletonizing procedure that scratches off unimportant edges. The skeletonizing algorithm adopted here uses the Medial Axis Transformation (MAT). The notion of MAT originally comes from the field of image processing, where it is used to compute the skeleton or medial axis of 2D shapes. By definition, the medial axis of a 2D image is the loci points, which are equidistant from at least two points along the region border (BLUM 1967). Figure 5.12 illustrates the effect of the MAT on the shape of two 2D images.

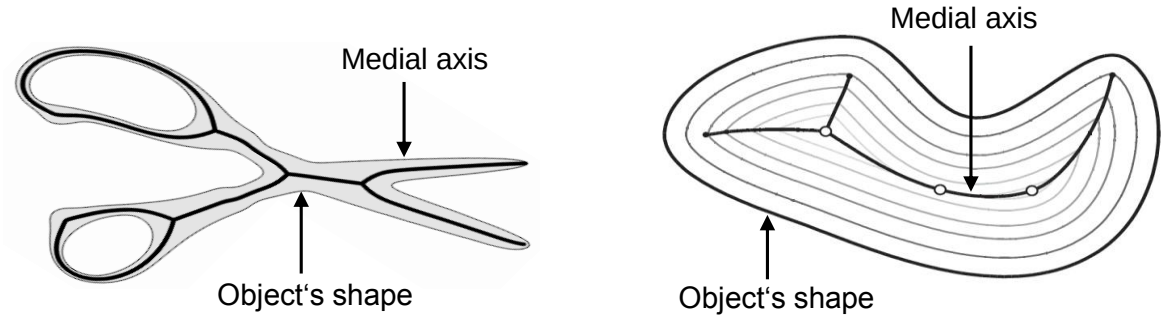


Figure 5.12: Effect of the medial axis transformation on 2D object's shapes (derived from ATTALI ET AL. (2009))

The MAT algorithm implemented in this work is inspired by the one proposed in ROESSL (2005 pp. 34–38). It consists of an iterative eroding process which peels off the patch's outer layer down to a set of piecewise linear curves that approximate the topology of the feature line. Before implanting the MAT, the patch is enlarged by a *region growing method* that reinserts some facets from those previously decimated by the *hysteresis filter*. Only those facets that share vertices with facets already contained in the patch are reinserted. Even though these new facets will in no way be part of the medial axes, they bring vertices along that are indispensable for a stable behavior of the MAT method elaborated below.

The MAT commences by iteratively characterizing the vertices according to their relevance to the patch in three classes: *center*, *disk*, and *boundary vertices* (see figure 5.13). *Boundary vertice*s are those that are on the patch border and can therefore not be part of the patch medial axes. They can and will be removed later on. A vertex is said to be a *center vertex* if none of its first-neighboring vertices belong to the patch's boundary. *Center vertices* are assumed to be part of the feature line and should therefore be retained. The neighbors of *center vertices* are labeled *disk vertices* and correspond to the vertices that would not bias the medial axis topography if removed. It is precisely here where the patch enlargement previously mentioned comes to bear. Had the discarded facets and, by extension, the boundary vertices not been reinserted, then too many vertices would have been eligible as *disk vertices* and consequently removed by the MAT. While this may not have

represented a big issue for large patches, those with narrow passages could have been completely swept away by the MAT, causing the complete contour extraction algorithm to fail.

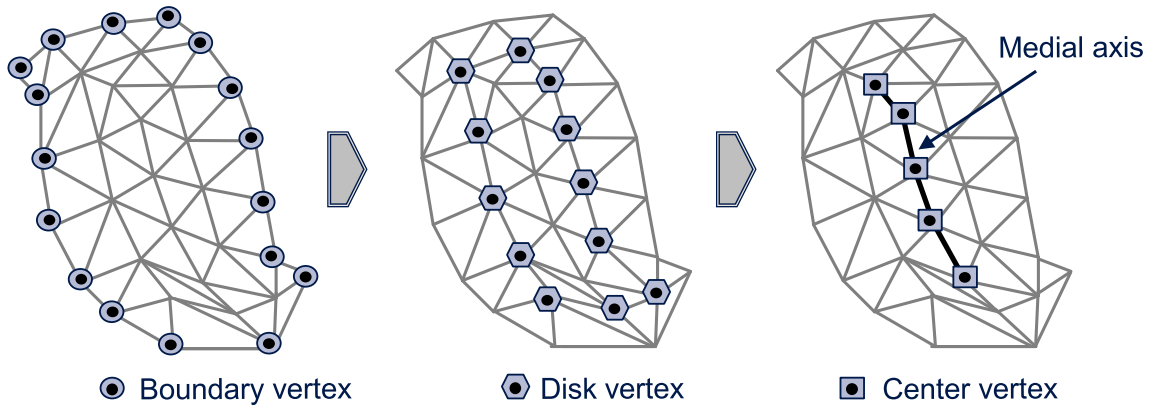


Figure 5.13: Pictorial illustration of the MAT function principle

Once the tagging of the vertices is concluded, vertices that are not labeled as center are simply deleted from the patch. From the center vertices left over, the patch skeleton is built by connecting them using the *nearest-neighbor principle*. The resulting skeleton may not exactly represent the original feature contour as specified in the CAD*;* nevertheless its topology will unequivocally resemble it. Figure 5.14 illustrates how the MAT method acts on the patches output of the preceding stage (see figure 5.11). All relevant design features are clearly discernable and their topology seems to be accurate. Furthermore, long feature contours are extracted without being fragmented, which is a strong indication of the robustness of the extraction algorithm.

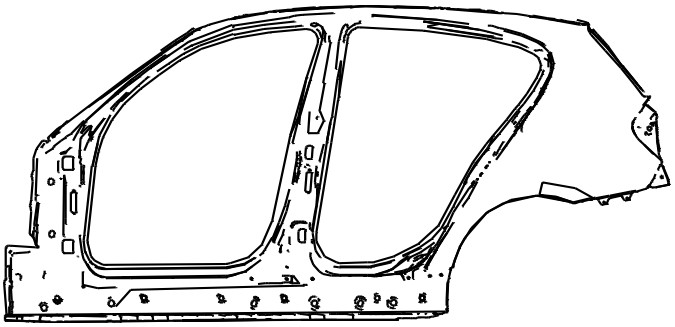


Figure 5.14: Results of the MAT applied on the edge patches isolated in the selection stage

Further experiments conducted on a variety of sheet metal parts showed similarly good results, as can be seen in figure 5.15.

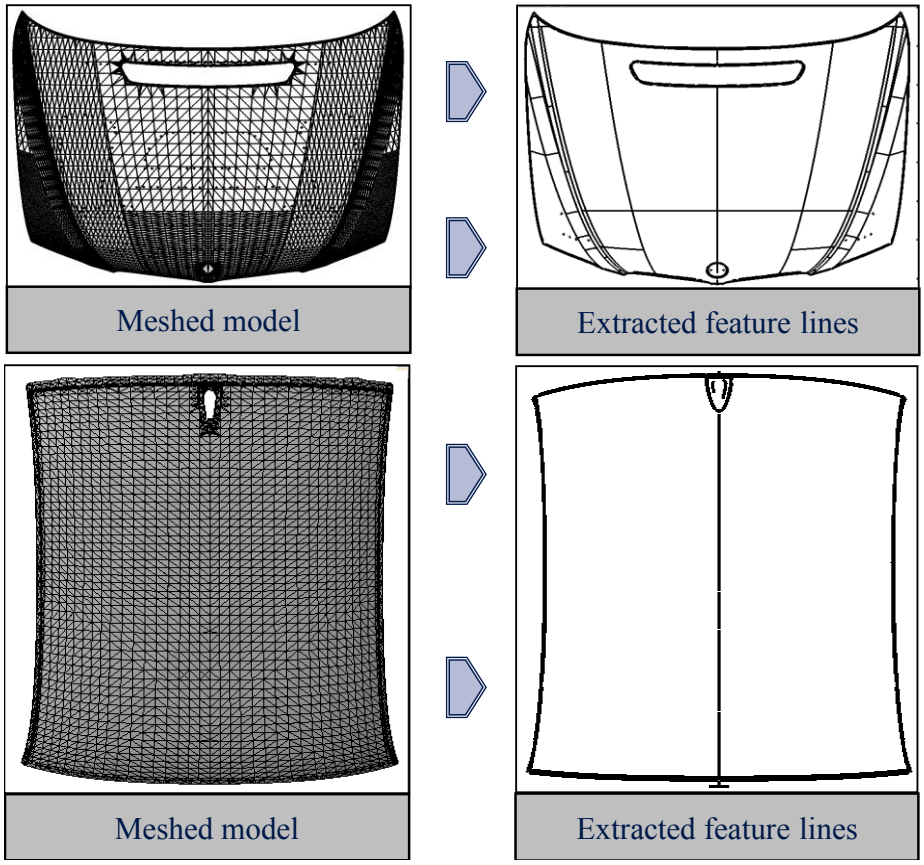


Figure 5.15: Outcome of the MAT algorithm on additional meshed models

5.3.2.7 Pruning Stage

The MAT algorithm devised above preserves details, but it is also very sensitive to tessellation noise. This fact is clearly recognizable in figure 5.14 by skeleton branches that are not morphologically significant. To handle this instability, a pruning filter is applied to the skeletons extracted in the previous phase. The filter denoises the skeleton by removing short and open skeleton segments containing fewer vertices than a threshold limit. Unfortunately, this simple and efficient filter has the disadvantage that this threshold limit has to be manually set. This is due to the fact that the amount of tessellation noise is unique to each CAD model.

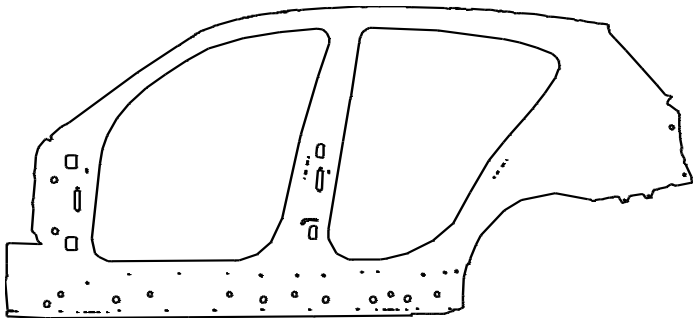


Figure 5.16: Noise-free feature contour lines

Hence, automatically tweaking this parameter is virtually unfeasible. Nevertheless, since the filter parameter is meant to be a positive integer, it is assumed that this task does not represent a challenge for a potential untrained operator. Figure 5.16 shows the result of the application of the pruning filter on the patches illustrated in figure 5.14.

5.4 Feature-based Task Description

Now that the feature lines have been singled out from the meshed model, they can be used to describe the digitization task at hand. To do so, the operator limits himself to just pointing and selecting one or more of the extracted features lines on a graphic screen. Once the features are selected, the operator has the option of specifying or not the orientation by which the sensor should approach the feature lines. If not, the orientation information encoded in the normal vector of each sharp edge is considered. It should be noted that by default, sharp edges are paired with a normal vector resulting from the interpolation method described in section 5.3.2.4. Advanced operators can also allocate totally new orientation to each of the normal vectors. Figure 5.17 illustrates a sheet metal part on which a digitization task is described with the method devised in this module. The left-hand side of this picture shows two selected feature lines, while the right-hand side displays the orientation of the normal vectors associated with the sharp edges in the selected feature lines.

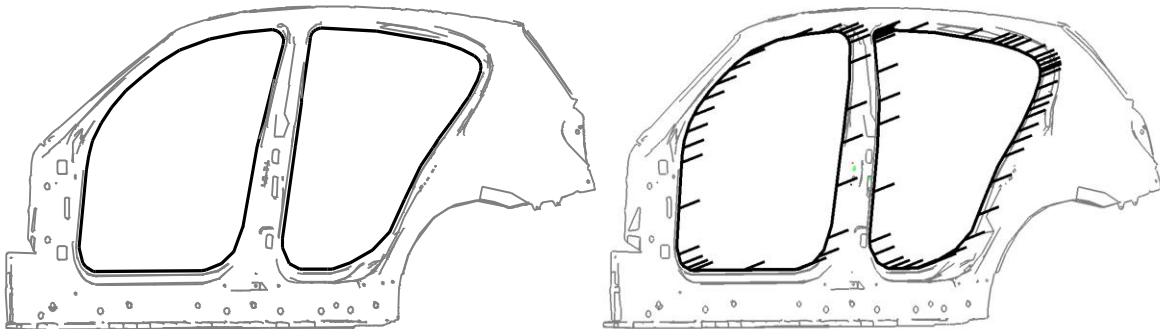


Figure 5.17: Feature-based description of the inspection task. The darked feature lines represent the choosen features.

As a closing statement for this chapter, it should be noticed that for non-complex WPs, an intervention of the operator for the description of the digitization task is not necessary at all. The entire feature set can be automatically considered as being part of a digitization task, rendering the manual step described above superfluous. But either way, the description of the digitization task can be performed by any employee, regardless of their exposure to technical details in the field of robotics and metrology—as long as there are familiar with PC environments.

6 Path and Trajectory Planning Module

6.1 Overview and Module Structure

The purpose of this module is to generate trajectories capable of guiding the sensor along the feature lines extracted in the task description module. The sensor trajectory problem is split into three hierarchical levels. The first level deals with the isolation of a solution space, which delimits the area in which adequate viewpoints to perform the digitization task can be searched for. The second level is devoted to the sketching of a method, which singles out adequate viewpoints and constructs a scan tour that leads the sensor through the viewpoint. The third and last level carries over the scan paths into scan trajectories steerable by the FIS.

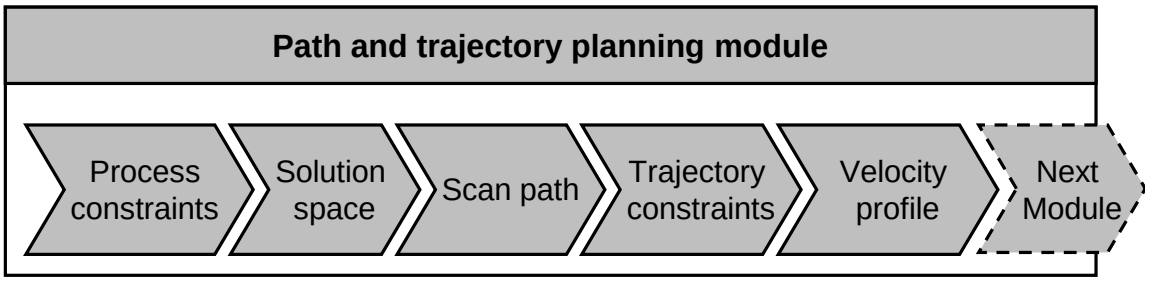


Figure 6.1: Module overview

Before proceeding with the elaboration of this chapter, it should be recalled that the inputs of this module are feature contours constituted by closely arranged points that are connected by straight-line segments. For convenience reasons, these points will henceforth be abbreviated as *TagPoint(s)*, which stands for target point(s).

6.2 Scan Process Constraints

For the feature line to be captured, sensor viewpoints are required that ensure a robust coverage of all its *TagPoints*. As stressed BY BERNARD & VÉRON (1999) and ELMARAGHY & YANG (2003), robust sensor coverage implies that at least the following constraints must be satisfied:

View angle constraint (*VAConstr*): This constraint requires that the angle of incidence θ between the incident laser beam and the surface normal at each *TagPoint* should be less than the limit view angle θ_{lim}. The latter is the maximum incidence angle by which the sensor can still capture the *TagPoint.* Ideally, the sensor should operate head on, that is *TagPoints* should be hit perpendicularly by the laser beam. In doing so, measurement noise can be drastically reduced (PRIETO ET AL. 2000).

Field-of-view constraint (*FoVConstr*): Another basic requirement is that a *TagPoint* being observed should lie in the active area of the camera. If this is not the case, then the *TagPoint* may be illuminated by laser beam but not collected onto the retina of the camera.

Visibility constraint (*VisConstr*): This constraint accounts for the bistatic nature of LSSs, i.e. the physical separation of the light-emitting source and the receiving optic. For this constraint to be met, neither the incident beam nor the camera view field should be shadowed during the measurement process.

Clearance constraint (*CleaConstr*): the sensor probe should not interfere with the WP or with any other component of the workcell during the scanning operations.

High-quality range image (*HiqualConstr*): To be suitable for QI the range image should be of high quality. What is meant by this will be detailed in section 6.5.4, where the *HiqualConstr* is to be merged into the sensor planning process.

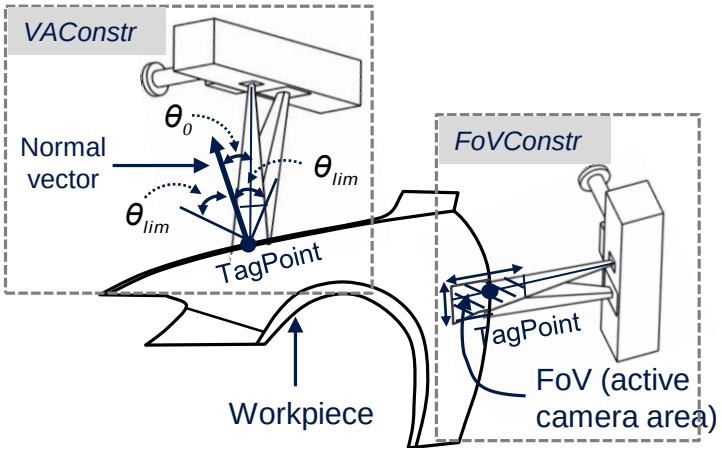


Figure 6.2: Schematic illustration of the VAConstr and the FoVConstr

For the sake of completeness it should not be forgotten that there are also several optical parameters that can significantly influence the digitization results. However, as stressed in section 3.5, it is assumed that the optical settings were already calibrated during the system commissioning phase so that there is no need to consider them at programming time.

6.3 Isolation of the Sensor Solution Space

6.3.1 Preliminaries

Similar to the visib*ility space*, the sensor solution space is meant to represent the portion of the *view space* from which the sensor can capture all the feature lines in

compliance with the constraints listed above, except the last one. The approach taken to computing it is based on quadratic *scannability cones* apexed at each *TagPoint,* as shown in figure 6.3. The triangulation principle the sensor is based on suggests the use of circular conical geometries to capture the portion of the space from which the sensor could theoretically collect a *TagPoint*. Successively applying the task constraints on each of those cones will reshape them in such a way that the view space around each *TagPoint* is reduced to a *scannability frustum.* As can be seen in figure 6.3, the *scannability frustum* exhibits the portion of the *scannability cone* from which the sensor's access to the corresponding *TagPoint* is guaranteed.

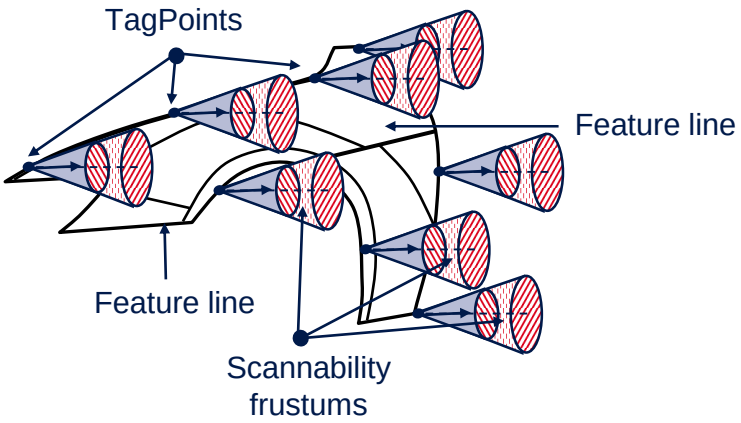


Figure 6.3: Solution space computation principle

6.3.2 Enforcing the *VAConstr*

The *VAConstr* is used in the calculation of the cone aperture angle α_{Cone} as follows: $\alpha_{Cone} = 2 \times \theta_{lim}$. Under optimal scan conditions, e.g. minimal light reflection, suitable laser intensity, θ_{lim} cannot exceed 60° and for shiny surfaces, 15° (FERNÁNDEZ ET AL. 2006). Thus, θ_{lim} leaves some room for combating optical disturbances such as surface reflectance caused by ambient illumination that can severely narrow the sensor working windows.

6.3.3 Enforcing the *FoVConstr*

To enforce the *FoVConstr*, a *scannability frustum* with an altitude equal to the sensor's DoFi is isolated at the cone base as shown in figure 6.3. Afterwards, the *scannability frustum* is pre-sampled in such a way that each sample viewpoint forces the sensor axis to pass through the *TagPoint*. The sampling scheme uses the following discretization parameters: azimuth φ_{az}, elevation φ_{el} and radius r. Figure 6.4 geometrically illustrates the meaning and the use of these three parameters. The discretization parameter φ_{az} is bounded between 0 and 360° to account for the circular geometry of the *scannability frustum*, φ_{el} is bounded between $\pm\theta_{lim}$ to

avoid violating the *VAConstr*. *r* is to be constrained within the upper and the lower limits of the DoFi to enforce the *FoVConstr*. From the trigonometric relationships in the *scannability cone*, the $[X\ Y\ Z\ \gamma\ \beta\ \alpha]$ parameters for each sampled viewpoint can be straightforwardly derived.

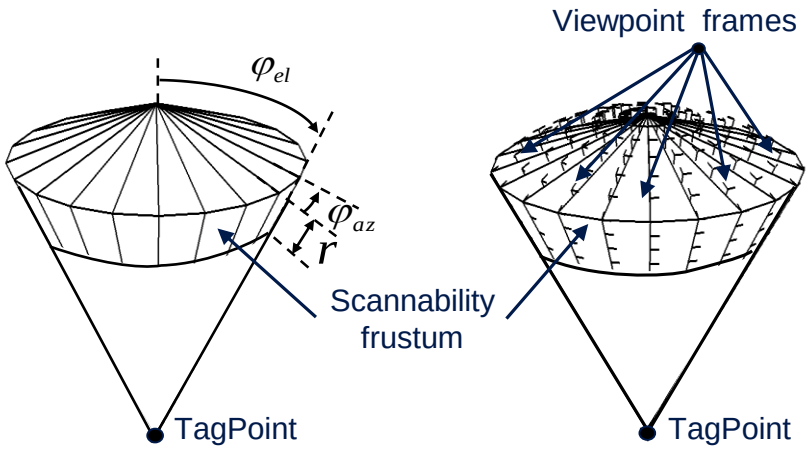


Figure 6.4: Discretization scheme of the scannability frustum (left), discretized scannability frustum (right)

6.3.4 Enforcing the *VisConstr*

Due to the bistatic nature of the sensor, occultation and shadow phenomena may arise at some *TagPoints*. The occultation happens when the *TagPoint* is hit by the laser beam but not seen by the camera (see figure 6.5 left) and shadowing occurs the other way around (see figure 6.5 right).

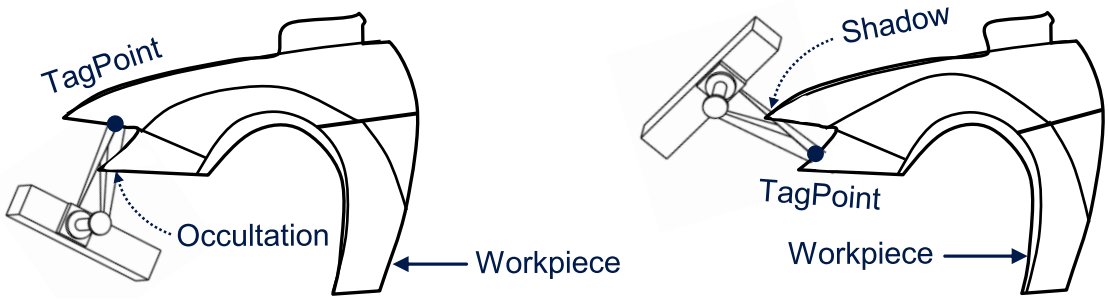


Figure 6.5: Occlusion Phenomena: a) occultation; b) shadowing

Referring to the *scannability cone* introduced above, it can be deduced that the *VisConstr* is fulfilled when, first, the *scannability cone* does not interfere with the WP. Additionally, the incident laser beam as well as the camera view frustum should lie entirely within the *scannability cone*, as shown in figure 6.6. Subsequently, the *VisConstr* is substituted by those two new constraints, which will be henceforth termed *VisConstr1* and *VisConstr2*.

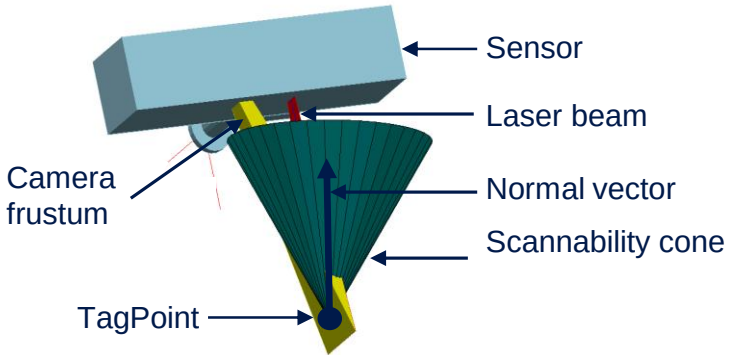


Figure 6.6: Sensor pose fullfilling the VisConstr2. Notice that the laser beam and the camera frustum are inside the cone. The scannability frustum is masked out for the pupose of better visualization.

To enforce the *VisConstr1*, the WP CAD and the cone CAD are tested for intersections using the collision engine. If they both interfere, this is an indication that some areas of the WP are cutting through the cone, thus shadowing some viewpoints in the *scannability frustum* as shown in figure 6.7 a). It is important to note that as the *scannability cones* are apexed on the WP at the *TagPoint*, there is always per se interference between both. However, since the collision engine cannot report multiple intersections between two objects, this unwelcome contact has to be eliminated. This is achieved by lifting up the *scannability cones* right before enforcing the *VisConstr1* and restoring the status quo ante once the collision tests are over.

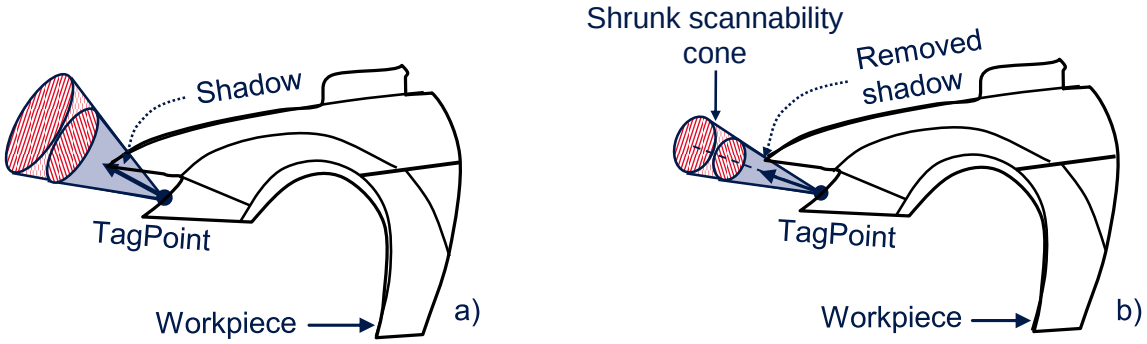


Figure 6.7: Inteference beween the workpiece and scannability cone

Each time that the collision query indicates interference between the WP and the cone, the latter is shrunk to remove this handicap (see figure 6.7 b)). The best way to reshape the cone volume is by tweaking the aperture angle α_{Cone}. In order to prevent the loss of too much volume, a procedure which finds the optimum cone aperture angle α_{opt} was designed that removes the collision handicap. This procedure is based on the dichotomy algorithm and operates by gradually narrowing down the searching range of α_{opt} until a desired precision is approached. Inputs of the algorithm include the minimum cone aperture angle α_{min} (≥ 0), the maximum

aperture angle α_{max} ($\leq \theta_{\text{lim}}$) and the desired precision of the approximation ε. Table 6.1 provides the pseudocode of the algorithm.

Table 6.1: Pseudocode of the dichotomy algorithm to optimize the cone aperture angle and thus indirectly the cone geometry

Algorithm: ***OptimizeConeApertureAngle***()
Input := $\alpha_{min} \geq 0$, $\alpha_{max} \leq \theta_{lim}$ and $\varepsilon > 0$
Output := α_{opt}
CheckConeCollision(α_{max}) if no collision, then $\alpha_{opt} = \alpha_{max}$ else (the solution α_{opt} must be between $[a, b]$) a = α_{min},b = α_{max}, $\varepsilon = 10^{-p}$ do compute the midpoint $\alpha_{mid} = (a + b)/2$ CheckConeCollision(α_{mid}) if collision, α_{opt} must be in the range $[a, \alpha_{mid}]$ $b = \alpha_{mid}$ else (no collision) α_{opt} must be in the range $[\alpha_{mid}, b]$ $b = \alpha_{mid}$ else (no collision) α_{opt} must be in the range $[\alpha_{mid}, b]$ $a = \alpha_{mid}$ end until $\lvert a - b \rvert \leq \varepsilon$ $\alpha_{\text{opt}} = a$, End return α_{opt}

Enforcing *VisConstr2* is much more straightforward than *VisConstr1*. To make sure that the camera frustum lies in the *scannability cone*, the cone aperture angle α_{opt} is further shrunk by the amount of the sensor triangulation angle δ. If the aperture angle α_{opt} happens to be smaller than the triangulation angle δ, then the previous step is not feasible and therefore is not carried out. Instead, a brute force algorithm that tests each viewpoint within the *scannability frustum* for shadowing comes into play. The shadowing testing mechanism relies on the ray-tracing algorithm the sensor simulation is based on and operates as follows: at each viewpoint, the scan simulation is activated and a bundle of primary rays are fired from the laser

source toward the *TagPoint*. As the *VisConstr1* is already fulfilled at this stage, those rays are expected to be free of occultation. Once those rays strike the WP surface, they are deflected towards the camera. If any of the deflected rays report an intersection on its way to the camera, it is assumed that the camera view frustum is obstructed, i.e. the *TagPoint* under investigation cannot be captured. As a consequence, the viewpoint is discarded from the *scannability frustum.*

6.3.5 Enforcing the *CleaConstr*

The constraint *CleaConstr* is accommodated by systematically positioning the sensor model at each viewpoint and querying the collision engine for interferences between the sensor and the WP. Viewpoints that fail the collision tests are simply unloaded from the solution space. After this step, the viewpoints left over in the *scannability frustum* comply with all the task constraints except the *HiqualConstr*, which will be considered later on.

6.3.6 The Task Solution Space

Systematically repeating the procedure above for each individual *TagPoint* of the feature lines allows for computing a local solution space for each of them. The union of all *scannability frustums* yields what can be called the *task solution space*. This represents the portion of the space in which a search can be made for candidate viewpoints that allow the sensor to capture the features under investigation. In finishing, it must be mentioned that since the viewpoints are described by their pose parameters $[X\ Y\ Z\ \gamma\ \beta\ \alpha]$ (see section 4.2.2), a frame is associated with each of them. The frame's unit vectors are labeled similarly to those of the sensor viewpoint e.g. $[\underline{\vec{s}}\ \underline{\vec{d}}\ \underline{\vec{a}}]$. The underscore under the vector names helps differentiate between the viewpoints' unit vectors and those of the sensor $[\vec{s}\ \vec{d}\ \vec{a}]$. With that said, for the sensor to visit a viewpoint, their corresponding unit vectors must be superimposed.

6.4 Viewpoint Selection and Path Planning

6.4.1 Preliminaries

Once the task solution space is delimited, a scan path consisting of an optimal sequence of viewpoints that lead the sensor through each *scannability frustum* of the task solution space is computed. An optimal sequence is one which minimizes and optimizes digitization data quality and process efficiency as well. Bearing in mind that exact solution to such problems are not tractable (see section 2.2.3.2), an appro-

ximation approach is adopted. The heuristic embodied in it is a composite algorithm that combines a path construction step, followed by a path improvement step.

6.4.2 Path Construction

This step is concerned with choosing suitable viewpoints from each *scannability frustum* and connecting them to form a scan path. Accordingly, the scan path can be abstracted as

$$ScanPath = \{ [X_{VP}\ \ Y_{VP}\ \ Z_{VP}\ \ \gamma_{VP}\ \ \beta_{VP}\ \ \alpha_{VP}]_i \qquad i = 1, \dots, n \} \tag{6.1}$$

The viewpoints are chosen depending on their contribution to the resulting length of the scan path. For efficiency reasons, it goes without saying that the scan path should be as short as possible. To tap into the vast amount of existing algorithms to solve the TSP, the scan path construction problem is reframed as follows: find a tour which takes the sensor through all *scannability frustums* and at the same time minimizes the path-length metric. To solve this TSP, a greedy[14] *nearest neighbor heuristic* was implemented. The algorithm proceeds by iteratively calculating the Euclidian distance between the present viewpoint and all those contained in the *scannability frustum* of the next *TagPoint*. The viewpoint with the shortest distance is then chosen and connected to the precedent viewpoint. Because there is no predecessor viewpoint for the entering *TagPoint*, i.e. the *TagPoint* in whose *scannability frustum* the first viewpoint is searched for, the nearest *neighbor algorithm* cannot be applied there. The entering viewpoint is selected in such a way that the sensor spots the *TagPoint* head on as much as possible. It should be noticed that the scan path computation procedure is based on the assumption that a collision-free straight path always exists between any pair of neighboring viewpoints. This can be considered true for sheet metal WPs since the viewpoints are most likely to keep the sensor at a safe distance, above the WP's surface.

6.4.3 Path Smoothing

Due to its straight-line fashion, the scan path generated in the previous step will presumably be characterized by cups and sharp turns (TSIANOS ET AL. 2007). Paths featuring such attributes can provoke mechanical stress, e.g. large accelerations and vibrations, in the robot actuators (NIKU 2011). Such unnecessary deficiencies may leave little chance to the FIS to steer the sensor along the scan path. Hence, the path curvature has to be lowered to a minimum, which is what this section is concerned

[14] A greedy heuristic creates a solution to an optimization problem by choosing at each stage a solution that offers the most obvious benefit—with the hope of finding the global optimum.

with. The proposed smoothing method draws on the elastic band framework. The crux of this framework consists in remodeling a path to be followed by a robot as an elastic rubber band (QUINLAN & KHATIB 1993). To clearly illustrate the rationale behind this method, a portion of the scan path specified by the viewpoints VP_1, VP_2, VP_3, VP_4 and VP_5 is considered in figure 6.8. Assuming that VP_2 *and* VP_4 are rigidly fixed, a rubber band stretched between VP_2, VP_3 and VP_4 will result in a force trying to pull and align VP_3 with VP_2 and VP_4. To mimic this physical behavior, artificial forces F_{23} and F_{43} are applied to VP_3. According to Hooke's law, these forces can be considered proportional to a scalar elasticity factor k_{path} and the Euclidian distance between VP_3 and its neighboring viewpoints VP_2 and VP_4. As can be seen in figure 6.9, the resulting force F_{Res} which acts on VP_3, will tend to pull the viewpoint VP_3 towards its equilibrium position VP_3', i.e. the position where F_{Res} vanishes. However, since VP_3' is clearly located outside of the *scannability frustum*, a further constraint is imposed on the elastic band algorithm, namely that of shifting the viewpoints only as far as they do not exit the *scannability frustum*.

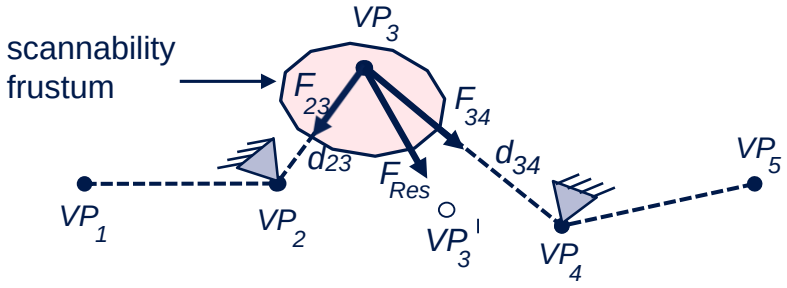


Figure 6.8: Function principle of the elastic band smoothing method. The scan path is meant to be viewed from the top. For the sake of clarity, only the scannability frustum to which VP_3 belongs is depicted.

Recursively applying this procedure to all viewpoints of a given scan path leads to a quite smooth path, as can be seen on figure 6.9. The smoothing procedure is repeated until a specific termination criterion assessing the benefit of continued searching is met. The termination criteria are the reaching of a pre-defined number of iterations or not having found any improvement within a given amount of iterations.

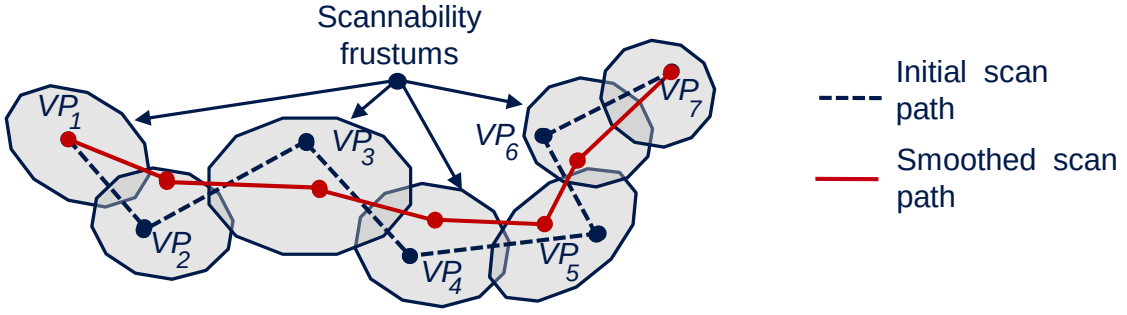


Figure 6.9: Effect of elastic band method on an artificial path (derived from

REINHART ET AL. 2008)

6.5 Scan Trajectory Constraints

6.5.1 Preliminaries

6.5.2 Overview

Now that scan paths are synthesizable, the next step aims at augmenting the latter with a temporal behavior to form a trajectory. However, this temporal behavior has strong repercussions on the manipulator motion and the quality of the digitizing results as well. Thus, before further elaborating on how the scan trajectories are generated, an insight will be given into the constraints to be accounted for during the trajectory generation process. These restrictions are imposed, on the one hand, by some of the manipulator physical restrictions. On the other hand, the high-quality range image constraint (*HiqualConstr*), introduced in section 6.2 but not yet implemented, also has a considerable role in shaping the scan trajectory. Figure 6.10 gives a brief overview of the trajectory generation mechanism.

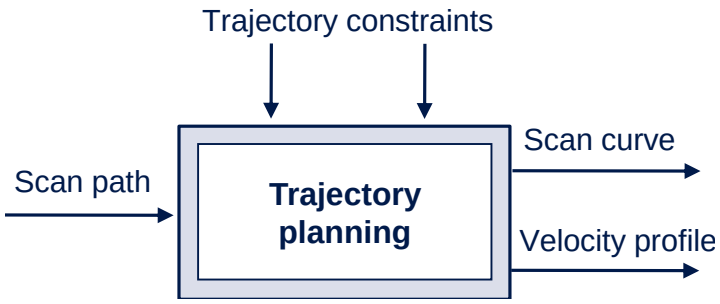


Figure 6.10: Principle of the scan trajectory generation mechanism

6.5.3 Smoothness Constraint

Despite the fact that scan paths are already been smoothed at the time of path construction, their piecewise linear fashion still represents a great handicap for an FIS. Indeed, piecewise linear paths cannot be executed by any motion devices without coming to rest at the transition point between the path segments (ETTLIN 2006 p. 144). Such stop-and-go patterned motions will inevitably induce jerky motions and occasion increased wear and vibrations on the manipulator's mechanism (CRAIG 2005 p. 203). From the process perspective, the main disadvantage of piecewise paths is clearly the increased process time. As stressed by CRAIG (2005 p. 202), trajectories driven by the manipulator's joints should follow a smooth continuous

motion pattern. This is particularly true for manufacturing processes for which the trajectory to be steered by the robot tool is critical (NOF 1999 p. 615). The trajectory of a robot joint is said to be smooth if it is C^2 *continuous*, i.e. the trajectory is continuous, it has a continuous first derivative (velocity) and its second derivative (i.e. acceleration) is continuous as well.

From this, it can be deduced that the scan trajectories should exhibit a C^2 *continuous* characteristic (*SmoothConstr*). The attentive reader may wonder if ensuring smooth scan trajectories in the task space will necessarily translate to soft trajectories in the manipulator joint space. The legitimacy of this question is strengthened in light of the fact that firstly, manipulator motion is controlled in the joints space and secondly, manipulator's physical limitations also arise in the joint space only. Fortunately, this question can be affirmatively answered. In ANGELES (2007 p. 235), appropriate evidence for this is provided. There it is emphasized that the inverse and forward kinematic equations, which map the manipulator end-effector position and orientation to joint values and vice versa, are continuous except when the manipulator traverses singular points. From this, it is inferred that any smooth trajectory planned in the task space is guaranteed to be smooth in the joint space, and the other way around, as long as the trajectory does not encounter a singularity.

6.5.4 Ensuring a High-Quality Range Image (*HiqualConstr*)

For a range image to be suitable for QI it must be "noise-free," "accurate," "dense," "homogenous," and "complete" (LARTIGUE ET AL. 2002). The first and second attributes are linked to the sensor characteristics (e.g. data sampling error, measuring accuracy) and cannot be tweaked by means of planning strategies. Regarding the remaining attributes, "dense" signifies that the distance between the sampled points is small, "homogenous" addresses the fact that the sampled points should be evenly spaced and "complete" stipulates that all targeted AoI are deemed to be represented in the measured range image. Acquiring a "dense" range image can be considered trivial, since the image granularity is mostly affected by the sensor resolution, which is configured only once at the phase of system commissioning. As for the range image's "homogeneousness" and "completeness", they are deeply dependent on the sensor trajectory and should therefore be explicitly accounted for in the sensor planning process.

As the sensor fires its laser at a fixed rate, a homogeneous range image can be achieved by driving the sensor with a constant velocity *(VelConstr)*. Here, it is not the velocity of the sensor carrier that is meant to be constant, but rather the velocity of the sensor focal point. A complete range image can be guaranteed by ensuring maximum coverage of the WP areas around the feature line by the laser strip. The

latter occurs when the scan axis crosses the feature line being measured *(CrossConstr)* with the sensor's slide vector being orthogonal to the scan trajectory *(OrthoConstr)*. For a better understanding of the notion of high-quality range image, two range images with different quality grades are schematically depicted in figure 6.11. The left half of this figure shows a range image obtained in abidance of the three constraints: *VelConstr*, *CrossConstr* and *OrthoConstr*. The right half, in contrast, displays a range image taken without enforcing these constraints. As a result, this range image is sparse and inhomogeneous.

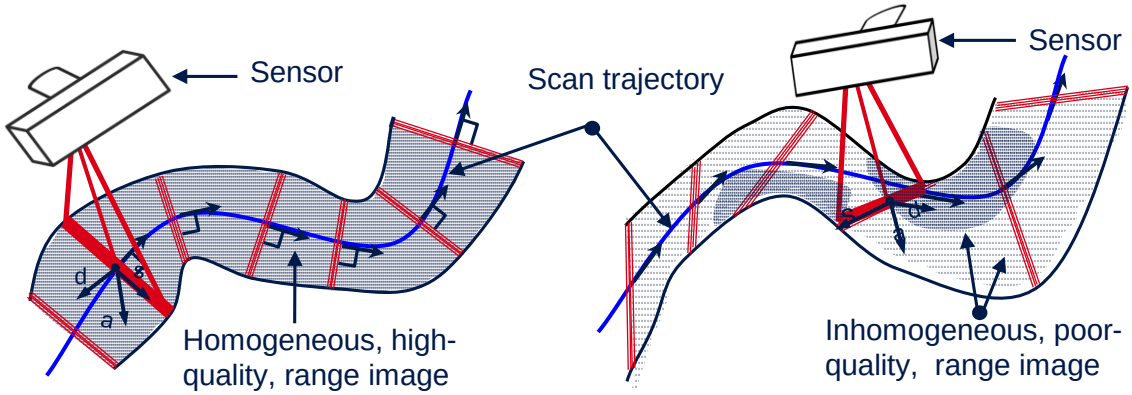


Figure 6.11: High-quality range image (left) versus poor-quality image (right)

6.6 Scan Trajectory Planning

6.6.1 Approach

Following standard practice in robotics, the laws of change of sensor position and orientation in the tasks pace with respect to time is determined in two steps (VERSCHEURE 2009). Firstly, the piecewise scan path is converted into a scan curve that passes through all the viewpoints' loci and simultaneously satisfies the *SmoothConstr*, *CrossConstr* and *OrthoConstr* constraints. Afterwards, the scan curve is augmented with a velocity profile that specifies the sensor motion in such a way that the *VelConstr* is fulfilled. Taken together, both of these elements are fully sufficient to unequivocally dictate the sensor trajectory (BIAGIOTTI & MELCHIORRI 2008 p. 425). Splitting the trajectory generation process this way holds the advantage that the sensor velocity can be freely modified without altering the geometric course of the scan curve. Moreover, to capitalize on the fact that the *CrossConstr* and *OrthoConstr* only affect the sensor orientation, the positional and rotational parts of the scan curve are treated separately, one at a time. This separation approach is rendered possible by the fact that manipulators are equipped with a spherical wrist without which it would not be possible (ANGELES 2007 p. 356).

Recalling that the scan path is actually composed of viewpoints that do not specify the pose of the sensor frame but rather that of the laser source frame F_{LS}, a shift operation which maps them to the sensor frame F_{FP} is imperative. The shift operation can be easily computed by considering the rigid relationship between F_{LS} and F_{FP} respectively (see figure 4.7). As a consequence, the scan path is reframed equivalently as a sequence of *n* (shifted) viewpoints through which the frame F_{FP} must past. On account of this, equation (6.2) can be rewritten as

$$ScanPath = \{ [X_{FP}, Y_{FP}, Z_{FP}, \gamma_{FP}, \beta_{FP}, \alpha_{FP}]_i \;\; i = 1, \dots, n \} \tag{6.2}$$

6.6.2 The Translational Scan Curve

6.6.2.1 Interpolating the Translational Scan Path

The translational scan curve is obtained by interpolating through the vectors $[X_{FP}]_i$, $[Y_{FP}]_i$ and of $[Z_{FP}]_i$ by means of piecewise polynomial functions. This procedure, which is well known as spline interpolation, has already established itself as more or less standard interpolation scheme (ANGELES 2007 p. 375). Of all types of splines, cubic splines are purported to be suitable for trajectory interpolation purposes (BRONSTEIN & SEMENDJAJEW 2001 p. 955). A significant reason for this is given by Holladay's theorem (HOLLADAY 1957). This theorem states that there is no another twice-continuous differentiable curve through the same data points that has less curvature than cubic splines. Hence, from Holladay's theorem it follows that using cubic splines to design the scan curve automatically enforces the *Smooth-Constr*. It is also interesting to note that cubic splines are the lowest-degree polynomial that allows continuity in velocity and acceleration (ELLERY 2000 p. 277). Due to all of these mathematical properties, spline-based trajectories display the following physical effects on manipulators:

- minimum path tracking error between the data points;
- minimum mean square acceleration and energy loss in the manipulator drive;
- low vibrational excitation of the drive system and manipulator structure.

6.6.2.2 Parametric Spline Representation

Interpolation techniques do not work on data points whose abscissa values backtrack or have loops in them. Thus choosing any of the vectors $[X_{FP}]_i$, $[Y_{FP}]_i$ or $[Z_{FP}]_i$ as abscissa values and considering the remaining two as the ordinate values in any interpolation method is not an option. The sequence of numerical values in those vectors may loop back, leading to a retrograde sensor motion when

only forward motion is expected. To circumvent this, the vectors of $[X_{FP}]_i$, $[Y_{FP}]_i$ and $[Z_{FP}]_i$ are each considered as ordinates values. The associated abscissa values for each of these vectors are determined by a cumulative chordale parameterization procedure through the viewpoints. Figure 6.12 illustrates the principle of the chordale parameterization on a curve.

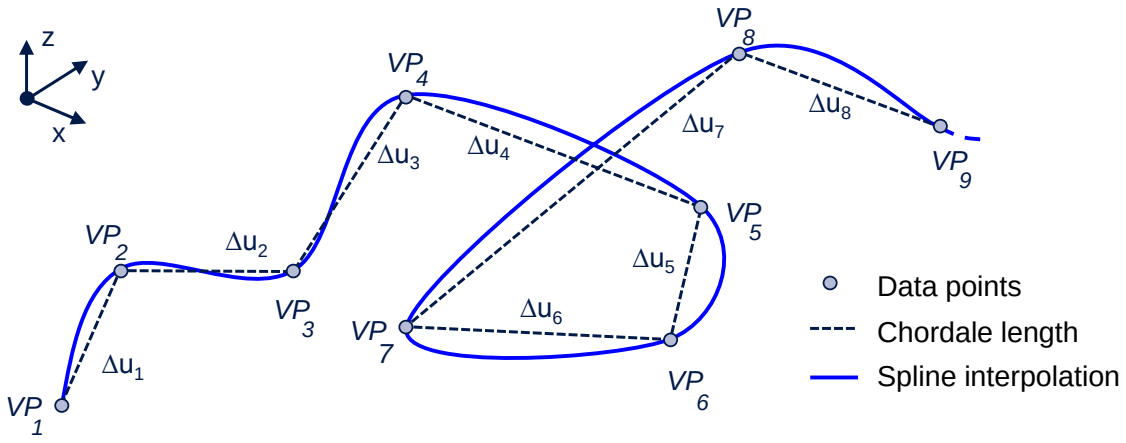


Figure 6.12: Principle of the cumulative chordale parameterization principle with respect to a spline interpolation

The chordale parameters u_i, corresponding to each via point $VP_i(x_i, y_i, z_i)$ are determined as follows:

$$u_k = \sum_{k=1}^{i} \Delta u_k \qquad with \;\; \Delta u_k = |VP_{k+1} - VP_k|, \quad i = 1, \ldots, n \tag{6.3}$$

with $|VP_{i+1} - VP_i|$ representing the Euclidian distance between VP_{i+1} and VP_i respectively and the parameter u_0 being assigned the value 0.

6.6.2.3 Interpolating the Viewpoints' Positions

In the following, the cubic spline interpolation through the viewpoints will be detailed for only one spatial direction of the scan path, e.g. the vector $[X_{FP}]_i$. The derivation of the two other directions is conceptually identical. Accordingly, for the moment, consider $P_{xk}(u_k, x_k)$ and $P_{x(k+1)}(u_{k+1}, x_{k+1})$, a pair of adjacent data points[15] along the x-dimension to interpolate through. A cubic interpolation polynomial $f_{xk}(x)$ between the subintervals $[u_k, u_{k+1}]$ can be written as

$$f_{xk}(u) \;=\; x_k = a_k(u - u_k)^3 + \; b_k(u - u_k)^2 + \; c_k(u - u_k) + d_k \tag{6.4}$$

[15] A data point consists of a chordale parameter and an u_k and a ordinate point x_k.

From that, the overall spline function S_x across the *n* via point pairs is given by

$$S_x(u) = \{ f_{xk}(u),\ u \in [u_k, u_{k+1}],\qquad k = 1, \dots, n\} \tag{6.5}$$

The coefficients $a_k, b_k, c_k,$ and d_k are determined by solving a linear system of equations obtained from the boundary conditions at the data point where the trajectory switches from one polynomial to another. To achieve continuity in position, velocity and acceleration of the sensor there, the following conditions must be met:

- each spline must cross the via-point at its extremity (interpolation);
- the velocity at the supporting point should be continuous;
- the acceleration at the supporting point must be continuous as well.

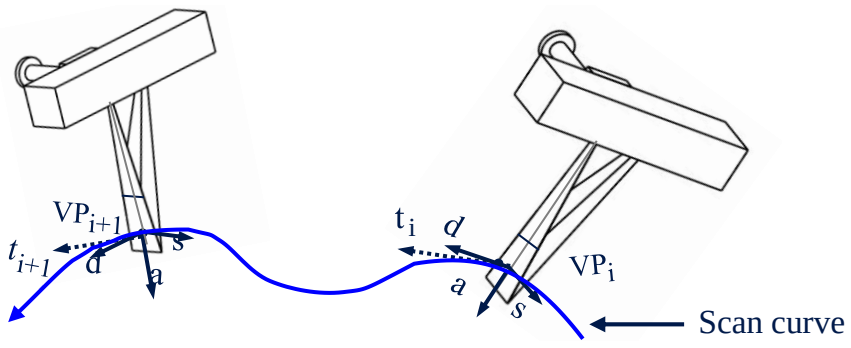
The starting point (u_0, x_0) and end point (u_n, x_n) of the whole trajectory are chosen to have velocity and acceleration equal to zero[16]. This implies that the sensor will start steering the trajectory from rest and will come to a full stop at its end. As it is straightforward, the derivation and solving of the system of equations resulting from these boundary conditions will not be further detailed.

6.6.3 The Rotational Scan Curve

6.6.3.1 Considering the *CrossConstr* and *OrthoConstr*

Preliminaries

As elaborated in section 6.6.1, the *CrossConstr* requires the sensor axis to cross the feature line being captured. This constraint is already fulfilled by each viewpoint since the discretization scheme applied on the *scannability cones* forces the scan axis to pass through the viewpoint's location (then referred to as *TagPoint*). The *OrthoConstr* imposes an orthogonal condition between the scanner's slide vector $\vec{s}$ and the path tangent vector. For illustrative purposes, two sensor poses whose orientations adhere to the *OrthoConstr* are depicted in figure 6.13.



[16] Splines with free boundary conditions are referred to as "natural splines."

Figure 6.13: Scan curve with two sensor poses that implement OrthoConstr (and the CrossConstr)

Seen from the viewpoint's perspective, the *OrthoConstr* requires all the viewpoints' frames to have the $\underline{\vec{s}}$-vectors orthogonal to the scan curve, i.e. to its tangent vectors. As the viewpoints' orientations are currently anything but related to the translational scan curve, series of rotations have to be applied on them to make them fulfill the *OrthoConstr.* Since these reorientation sequences are to be performed locally, a special reference coordinate system is needed at each viewpoint. For reasons of convenience, one of the unit vectors of this coordinate system should be tangential to the translational scan curve.

The Accompanying trihedron

From differential geometry, it is known that a triad of mutual orthogonal vectors—e.g. tangent e_t, normal e_n and binormal $e_b$vectors—can be per se associated with every point of a smooth curve (SICILIANO & KHATIB 2008 p. 204). While e_t and e_n are directly tied to the course of the curve, e_n is simply the cross product of the other two (see figure 6.14 a). These vectors span a right-handed frame, commonly referred to as the Frenet-Serret[17] triad or the accompanying trihedron (ATH). Since the unit vector e_t of the ATH is by definition always tangential to the curve, it can act as the local reference frame sought. Paraphrasing the Frenet-Serret theory, an ATH is tied at each viewpoint location along the translational scan curve. The ATH's unit vectors are easily calculated according to the Frenet-Serret theory, as shown in the following equations.

$$e_t(u) = \left(\frac{S'_x(u)}{|S'_x(u)|} \quad \frac{S'_y(u)}{|S'_y(u)|} \quad \frac{S'_z(u)}{|S'_z(u)|}\right)^T \tag{6.6}$$

$$e_n(u) = \left(\frac{S'_x(u) \cdot S''_x(u)}{|S'_x(u) \times S''_x(u)|} \quad \frac{S'_y(u) \cdot S''_y(u)}{|S'_y(u) \times S''_y(u)|} \quad \frac{S'_z(u) \cdot S''_z(u)}{|S'_z(u) \times S''_z(u)|}\right)^T \tag{6.7}$$

$$e_b(u) = e_b(u) \times e_t(u) \tag{6.8}$$

where S'_x and S''_x stand for the first and second derivatives of the spline along the x-coordinate. The meaning of the remaining terms is to be understood analogously. Combining the ATH's unit vectors as shown in equation (6.9) leads to a homogeneous transform $\mathrm{F_{ATH}(u)}$, describing the ATH with respect to the fixed WP frame.

[17] Named after two French mathematicians who independently devised them: Jean Frédéric Frenet, in his thesis of 1847, and Joseph Alfred Serret in 1851.

$$F_{ATH}(u) = \begin{bmatrix} e_t(u) & e_n(u) & e_b(u) & [s_x(u), s_y(u), s_z(u)]^T \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.9)$$

As can be seen in frame a) of figure 6.14, the Frenet-Serret frame may have its $e_b(u)$ vector pointing upwards depending on the torsion and curvature of the curve. Indeed, the vector $e_b(u)$, and by extension the vector $e_n(u)$, will experience sudden reversals upon passing through inflections, which will make them point in the opposite direction to which they were pointing before the inflection point. However, with regard to the subsequent steps, it is more convenient to have an ATH whose binormal vector $e_b(\mathrm{u})$ always points the same direction than the corresponding viewpoints' approach vector $\vec{a}$, that is to say, pointing downwards. In order to detect such a direction reversal, the scalar product between both vectors at each viewpoint is monitored. A negative scalar product suggests that $e_b(\mathrm{u})$ is pointing upwards. In this case, the direction $e_b(\mathrm{u})$ is rectified by spinning the ATH around the e_t vector by an angle of 180°, as shown in frame b) of figure 6.14.

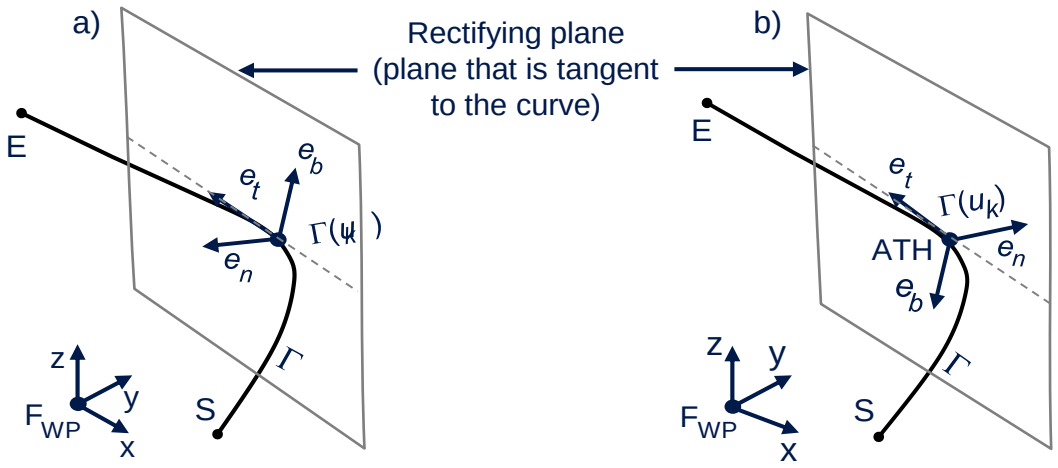


Figure 6.14: a) Frenet-Serret frame along a parametric curve; b) Accompanying trihedron (ATH) obtained through a 180 spin along the vector $e_t(u)$

Enforcing the *OrthoConstr*

Now that each viewpoint location is assigned an ATH, they can be properly reoriented to have their $\vec{s}$-vectors at right angles to the corresponding ATH's $\vec{e}_t$-vector. It should be noted that when a viewpoint's, $\vec{s}$-vector is perpendicularly aligned to the curve tangent plane this does not necessarily mean that the remaining unit vectors must also be superposed pairwise. To better explain this, consider $\vec{s}'$ $\vec{d}'$ and $\vec{a}'$ being the unit vectors of a viewpoint at which the sensor is intended to satisfy the *OrthoConstr* (see figure 6.15). From the *OrthoConstr* it is obvious that the vector $\vec{s}'$ is perpendicular to $\vec{e}_t$. As a consequence, $\vec{d}'$ is contained in the tangential plane spanned by $\vec{e}_t$ and $\vec{e}_b$. However, there are still two rotational DoF,

which prevent $\vec{d}'$ and $\vec{a}'$ being collinear to $\vec{e}_t$ and $\vec{e}_b$, respectively.

Having said that, no matter how arbitrarily a viewpoint frame ($\vec{s}'$ $\vec{d}'$ $\vec{a}'$) is oriented with respect to the corresponding ATH, its vector $\vec{s}'$ can be aligned orthogonal to the ATH tangential vector e_t by an elementary rotation around the vector $\vec{a}'$. Instead of proceeding this way, a more straightforward approach is adopted. It consists of directly determining the end state of the viewpoint frame as it should be in order to fulfill the *OrthoConstr*.

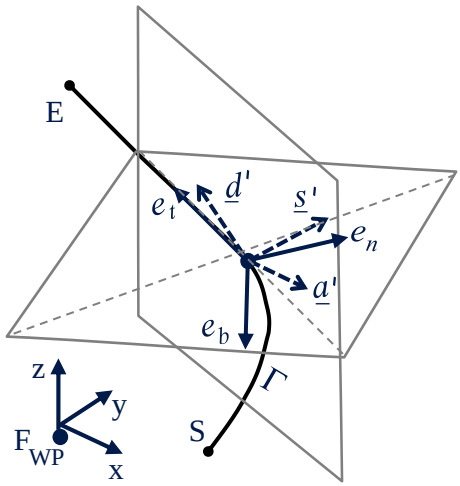


Figure 6.15: Illustration of a viewpoint frame ($\vec{s}'$ $\vec{d}'$ $\vec{a}'$) enforcing the OrthoConstr

From the assertion that proceeding through a single rotation would leave the vector $\vec{a}$ invariant, it can be deduced that $\vec{a}$ is identically equal to $\vec{a}'$. Since $\vec{s}'$ is by definition orthogonal to $\vec{a}'$ (($\vec{s}'$ $\vec{d}'$ $\vec{a}'$) is an orthogonal frame) it can be inferred by extension that $\vec{s}'$ is also perpendicular to $\vec{a}$. Because the vector $\vec{s}'$should be simultaneously perpendicular to $\vec{e}_t$ (*OrthoConstr*), it can be determined by means of the following cross product:

$$\vec{s}' = \vec{a} \times \vec{e}_t \tag{6.10}$$

The last unit vector $\vec{d}'$ must be laid out in such a way that the three vector $\vec{s}'$ $\vec{d}'$ and $\vec{a}'$ form a right-handed orthogonal frame. This is done by cross multiplying the $\vec{s}'$ and $\vec{a}$ (or $\vec{a}'$) as shown below

$$\vec{d}' = \vec{s}' \times \vec{a} \tag{6.11}$$

Figure 6.16 shows the application of the *OrthoConstr* to a sequence of viewpoints and their associated sensor poses, respectively. As can be seen on the left frame of the figure, the sensor poses are arbitrarily oriented before the enforcement of the *OrthoConstr*. After the *OrthoConstr* is applied to them, all sensor poses are now

orthogonal to the scan curve. A favorable side effect of this reorientation of the viewpoints is that they are now rearranged in a way that minimizes the reorientation of the sensor from one to the next. This is clearly apparent from the concordant alignment of the sensor shafts depicted on the right-hand side of figure 6.16, as compared to the completely disorganized sensor shafts pictured on the left. It appears that without the reorientation induced by the *OrthoConstr*, the viewpoint sequence as depicted on the left-hand side of figure 6.16 would not have been practicable by the FIS. The reason for this being that there is little chance that some of the viewpoints would be reachable without the FIS colliding with the WP.

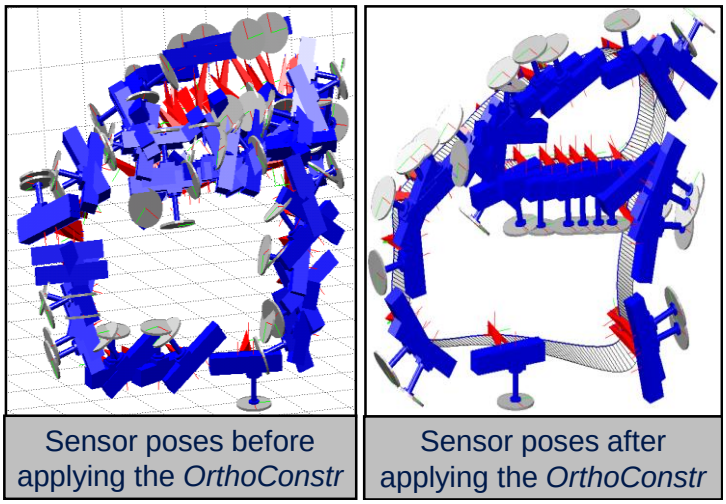


Figure 6.16: Effect of the OrthoConstr on disordered sensor poses

Nevertheless, through the orientation, some viewpoints may end up violating the scan constraints they once fulfilled before the enforcement of the *OrthoConstr*. As the rearrangement can be restrained to a rotation around their vector $\vec{a}$, it can be deduced that the new viewpoints poses still comply with the *VAConst*, *FoVConstr* and *VisConstr*. As regards the *CleaConstr*, it is not altogether impossible that some reoriented viewpoints may occasion collisions between the sensor probe and the WP. Hence, each viewpoint on the scan path is again tested for collision in accordance with the procedure specified in section 6.3.5. If a collision is detected, a rollback algorithm tries to escape this handicap by incrementally rotating the viewpoint back towards its initial orientation before the enforcement of the *CleaConstr*. An incremental rotation step of 3° has turned out to be an acceptable trade-off between computation time and the closest possible preservation of the *OrthoConst*. After each incremental rotation, the intermediary sensor viewpoint is re-queried for collision. As soon as a collision-free intermediate viewpoint is identified, the rollback algorithm exits and the intermediary viewpoint is substituted for the viewpoint that engendered the collision at the first place. This intermediary viewpoint will not fulfill the *OrthoConst*, but at least it represents a good compromise for avoiding a

deadlock of the sensor trajectory planning module.

6.6.3.2 Interpolating the Viewpoints' Orientations

Recalling the elaboration of the translational scan curve, one can logically expect that a spline interpolation through the viewpoints' orientation vectors $[\gamma_{FP}]_i$, $[\beta_{FP}]_i$ and $[\alpha_{FP}]_i$ will guarantee smooth orientation changes along the translational the scan curve. Another advantages of proceeding that way is that the angular velocities (orientation change rates) and acceleration (velocity change rates) throughout the translational scan curve will be inevitably smooth (BIAGIOTTI & MELCHIORRI 2008 pp. 348–349).

Subsequently, the interpolation procedures elaborated to obtain the translational scan curves is transferred to the viewpoints' orientation vectors to obtain the rotational scan curve. However, now that the viewpoints have been reoriented to fit the *OrthoConst*, the old orientation values ($\gamma_{FP}\ \beta_{FP}\ \alpha_{FP}$) calculated at the path planning stage are no longer valid[18]. A new orientation angle ($\hat{\gamma}_{FP}\ \hat{\beta}_{FP}\ \hat{\alpha}_{FP}$) relative to the WP must be calculated instead. The following procedure is utilized for this purpose: first, the ATH orientation matrix R_{ATH}, which is known to be expressed in the WP frame, is determined at each viewpoint. As equation (6.12) shows, calculating R_{ATH} is obvious, as this matrix is already embodied in the first three columns of the ATH's transformation matrix, derived in equation (6.9).

$$R_{ATH} = [e_t \quad e_n \quad e_b] \tag{6.12}$$

Thereafter, the relative orientation of each viewpoint with respect to its associated ATH frame is determined. Since the ATH tangent vector e_t and the viewpoint slide vectors $\underline{\vec{s}}'$ are at a right angle, one rotational DoF between the both frames is already locked. This results in each viewpoint frame being carried into the associated ATH frame by two elementary rotations: first, a rotation $R(\psi)$ around $\underline{\vec{s}}'$ aligning $\underline{\vec{d}}'$ along e_t and second, a rotation $R(\phi)$ around the current $\underline{\vec{d}}'$ superposing both frames. The combined rotation matrix which expresses the viewpoint frame's orientation with respect to the WP frame is therefore obtained by pre-multiplying R_{ATH} by $R(\psi)$ and $R(\phi)$ as follows

$$R_{VP} = R(\psi) \cdot R(\phi) \cdot R_{ATH} \cdot \tag{6.13}$$

From the orientation matrix R_{VP}, the roll, pitch and yaw rotation's angle ($\hat{\gamma}_{FP}\ \hat{\beta}_{FP}\ \hat{\alpha}_{FP}$), which express the viewpoint orientation relative to the WP frame, can be derived using the inverse solution for Euler's angle (rotation sequence ZYX).

[18] The reorientation leaves the viewpoints' positions $[X_{FP}, Y_{FP}, Z_{FP}]_i$ intact.

A second look at equation (6.13) reveals that it is much more advisable to directly interpolate the viewpoint relative orientation $[\psi]_i$ $[\phi]_i$ to end up with a similar solution than direct interpolation of $[\hat{\gamma}_{FP}]_i$, $\left[\hat{\beta}_{FP}\right]_i$ and $[\hat{\alpha}_{FP}]_i$. Indeed, since the ATH is directly related to the translational scan curve, which is known to be smooth, then it can be assumed that the orientation changes of the ATH along the scan curve are inevitably smooth. Therefore it is sufficient for only the relative orientation change between the ATHs and the viewpoint frames to be smooth in order to keep the combined rotation changes smooth. Based on this insight, only the vector of relative orientations angles $[\psi]_i$ and $[\phi]_i$ are interpolated. To keep the positional and orientation scan curves coordinated, the chordale parameters u_i derived for interpolating the former are also applied to the orientation interpolation. As the spline interpolation procedure implemented in this research has already been extensively treated in section 6.6.2, it is omitted at this juncture to avoid repetition.

6.6.4 Adding the Cartesian Velocity Profile to the Scan Curve

6.6.4.1 Preliminaries

To entirely define the scan trajectory, each scan curve has to be assigned a motion law that specifies the modality by which it must be tracked by the sensor. The motion law may be specified either by the time necessary to track the entire scan curve or the velocity by which the sensor progresses across the scan curve. From the *VelConstr*, it is known that this motion law should impose a constant velocity on the sensor. However, specifying the cycle time does not guarantee a constant tracking velocity. Hence, specifying the velocity profile along the whole scan curve would appear to be the more meaningful approach. Expressing the velocity control law along the scan curve is equivalent to the parameterization of the velocity profile with respect to the curve's arc length. However, since the relationship between the chordale parameter and the arc length is highly nonlinear (see equation (6.5), the scan curve must be re-parameterized according to the arc length.

6.6.4.2 Natural Parameterization of the Scan Curve

A curve in space is said to be naturally parameterized when the arc length of the curve serves as the curve parameter. Thus, every space curve possesses a natural parameterization even though it may be nontrivial to compute the arc length in practice. In order to re-parameterize the scan curve, an unequivocal mapping between the arc length, denoted here by s, and the chordale parameter u must be achieved. From differential geometry it is known that the arc length to be found can

be analytically expressed by the geometric integration

$$s(u_{k+1}) = \int_{u_k}^{u_{k+1}} \sqrt{S_x'(u)^2 + S_y'(u)^2 + S_z'(u)^2}\, du \qquad \text{with } s(u_0) = 0 \quad (6.14)$$

where $S_x'(u)$, $S_y'(u)$ and $S_z'(u)$ denote the scan curve first derivatives. As equation (6.14) clearly shows, it is clear that a bijective relationship exists between the chordale parameter u and the natural parameter s. Hence the former can be readily substituted by the latter in the scan curve equation (6.5), which is equivalent to re-parameterizing the scan curves with respect to s.

Nevertheless, a closed-form expression for equation (6.14) only exists for quite simple functions of low degree, at most two (FIUME 1995). Therefore a numerical approach is adopted for the scan curve whose spline polynomials are of degree three. As the chordale lengths $(\Delta u_1 \cdots \Delta u_n)$ have already been computed (see section 6.6.2.2), the trapezoidal Newton-Cotes[19] approximation technique lends itself to this endeavor. Once $s(u_k)$ is calculated at each discrete value u_k, the knot pairs $(u_k, s(u_k))$ are interpolated through a cubic spline. This yields a strictly monotone and continuous curve that cross link the chordale and natural parameters.

6.6.4.3 Implementing the Cartesian Velocity Distribution Profile

As stressed in MACFARLANE & CROFT (2003), in order to accommodate the physical limitations of a robot manipulator, the velocity profile should comprise:

- a ramp up segment to increase the velocity from zero to the nominal scan velocity V_0;
- a cruise segment where the nominal velocity is maintained constant;
- a ramp down segment to decrease the velocity back to zero.

From this, it is clear that the scan velocity should track a trapeze-like profile such as the one depicted in frame a) of figure 6.17. The duration of the acceleration and deceleration segments are parameterized and must be tweaked with respect to the manipulator's characteristics. Unfortunately, a pure trapezoidal velocity motion profile presents discontinuous acceleration jumps, as shown in frame b) of figure 6.17. It is also characterized by an impulsive jerk[20] profile, as illustrated in figure 6.17 frame c). BIAGIOTTI & MELCHIORRI (2008 p. 79) stress that such a motion law may result in undesirable vibrational effects as well as increasing wear and tear in the

[19] The Newton-Cotes formula, named for Isaac Newton und Roger Cotes, is a family of numerical integration techniques geared toward approximating an integral using interpolating polynomials of degree 1 (Trapezoid formula), degree 2 (Simpson's formula), degree 3 (Simpson's 3/8 formula) or degree 4 (Boole's formula).

[20] Jerk refers to the first derivative of an acceleration curve.

manipulator's joints.

Borrowing from the notion of a bell-shaped velocity profile by BIAGIOTTI & MELCHIORRI (2008 pp. 79–82), the velocity profile of the scan sensor that was intended to be trapezoidal is smoothed by adding a parabolic blend at the beginning and the end of the acceleration and deceleration segments in figure 6.17 frame d). In this manner, the acceleration profile becomes continuous and jerk values are bounded as well (seen figure 6.17, frames e) and f)). The pay-off is less stress in the manipulator joints and smooth motion characteristics.

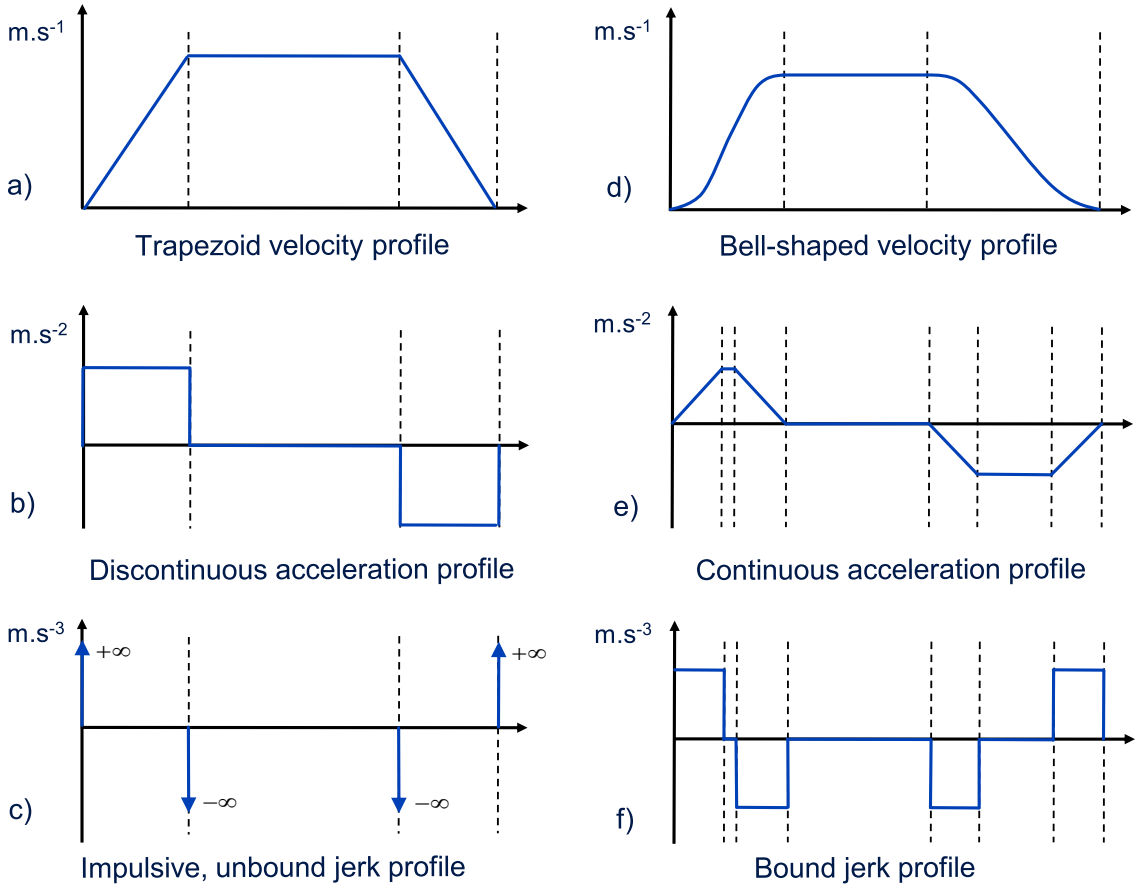


Figure 6.17: Comparison of a trapezoidal velocity profile with linear ramp up and down against a bell-shaped trapezoid profile with parabolic blends

In some cases it may happen that a scan trajectory cannot be driven in one stroke by the manipulator. This is the case, for instance, when a change of the FIS configuration is necessary which implicates that the manipulator temporarily withdraws from the scan curve. To handle this issue, so-called stop points, where the FIS comes to rest before initiating the configuration change, have to be inserted along the scan curve.[21] To do so, the velocity profile is augmented with further accele-

[21] The issue of configuration changes is comprehensively treated in section 7.3

ration and deceleration segments. Figure 6.18 shows such a velocity profile entailing a series of acceleration and deceleration phases to accommodate the start-and-stop motion induced by the insertion of five stop points.

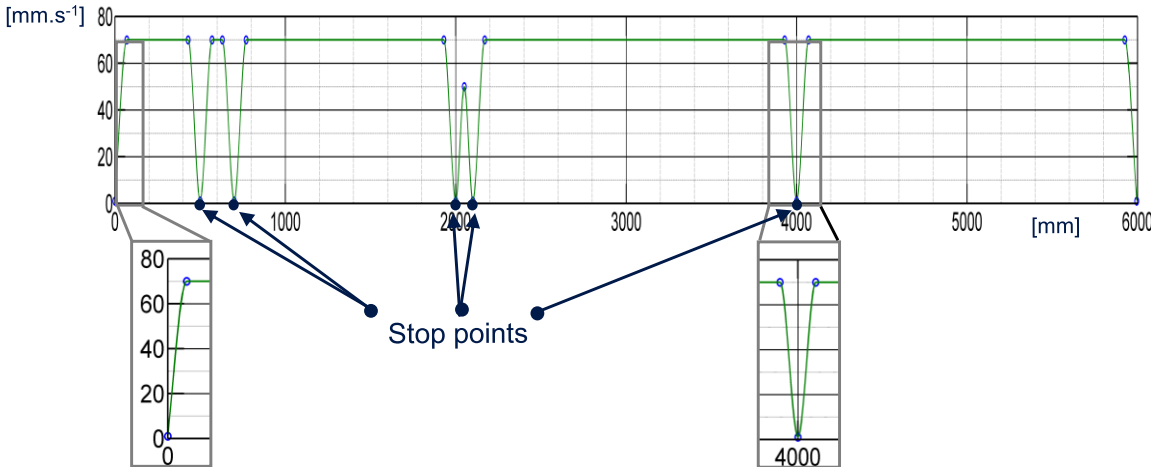


Figure 6.18: Velocity profile augmented with stop points

6.6.5 Deriving the Angular Velocity Profile

Using the Cartesian velocity profile, the time frame for the sensor to drive between two arbitrary viewpoints is fully determined. Within the same time frame, the sensor must also have overcome the orientation difference between the two viewpoints. From this, it is clear that the Cartesian and the rotational velocity are implycitly mutually linked. Therefore, defining one of these two velocities is sufficient to fully characterize the sensor motion. However, it is self-evident that a human operator will have difficulties specifying the sensor's angular velocity. Therefore, specifying the translational velocity is more user-friendly, which is why this method was favored in this study.

6.6.6 Illustrative Example

To underscore the performance of the path and trajectory planning module, an illustrative example will be given of scan trajectories generation based on the feature contours highlighted in frame a) of figure 6.19. The step-by-step results from running the algorithm on this feature contour are outlined in the remaining frames of the same figure. Frame b) represents the task solution space, while frame c) shows the scan path optimizing the path length metric. In frame d), the smoothed scan path (solid lines) is superimposed onto the unsmoothed path (dashed line) to emphasize the improvement realized in terms of lesser curvature. Frame e) displays the translational curve. In frames f) and g), the orientation of the sensor poses are

illustrated before and after the application of the *OrthoConstr*. Finally, frame h) displays the shape of the orientation scan curve, while frame i) depicts the simulated point range image obtained by steering the virtual sensor along the scan trajectories.

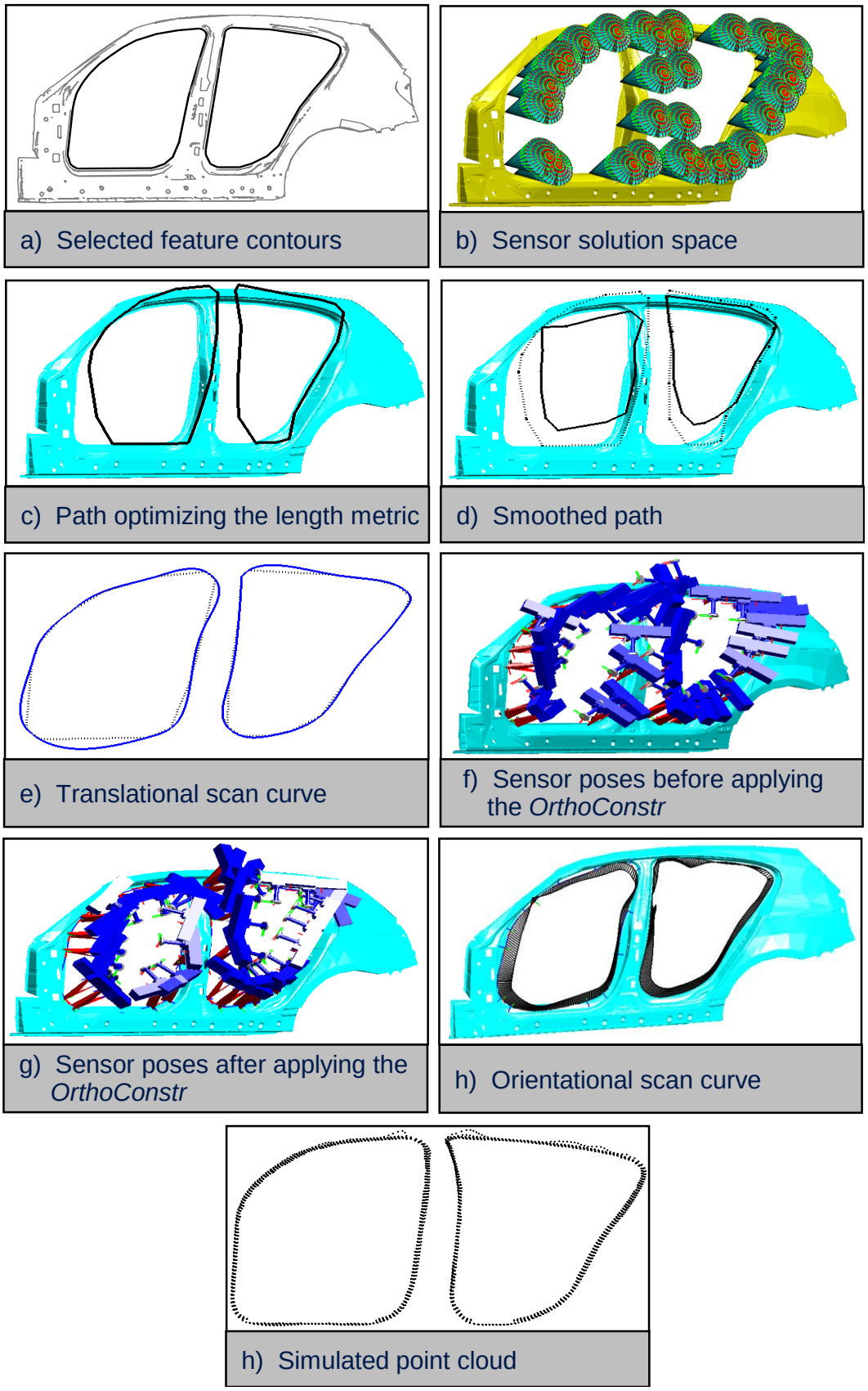


Figure 6.19: Example demonstrating the step-by-step results generated by the path and trajectory module

7 Motion Planning Module

7.1 Overview

Before scan trajectories are handed over to the real FIS for execution, they have to undergo further processing steps that aim to ensure that they are in keeping with the FIS's physical restrictions. Therefore, this chapter is concerned with the problem of finding which motion a specific manipulator should perform to steer the sensor along an automatically planned scan trajectory. To this end, the scan trajectory will be augmented with further constraints that fix ambiguousness embodied in it. Scan trajectories which may cause the FIS to transgress its physical limits will be locally modified to suppress this handicap. In addition, so-called transfer motion, which allows the FIS to navigate from its rest posture to the WP and back, will be automatically generated. This chapter concludes with a visual servoing-based system to cope with the mismatch between the virtual and real FIS workcells.

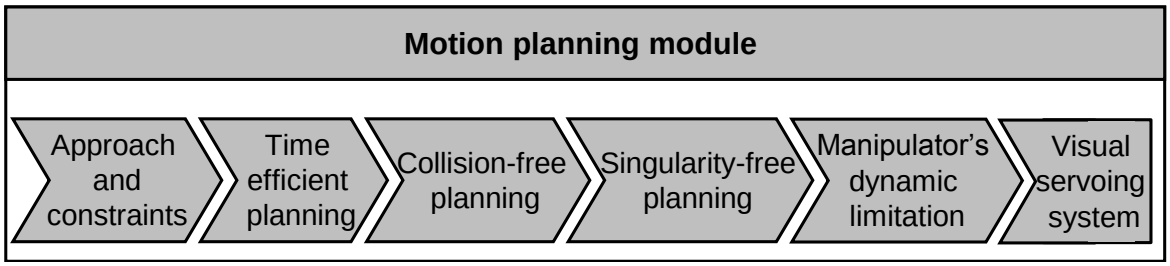


Figure 7.1: Module overview

7.2 Approach and Constraints

Motion planning can be defined as finding a path for a kinematic device from a given start to a given goal posture within its workspace without violating some predefined constraints (GERAERTS & OVERMARS 2006). Extensive treatment of the different aspects of constrained robot motion planning can be found in LATOMBE (1991) and LAVALLE (2006). However, for the purpose of this research it is enough to consider the following constraints:

- time efficiency constraints;
- kinematic constraints in term of collision-free and singularity-free motion;
- dynamic constraints in term of torque limits.

The planning activities that enforce these constraints will be carried out at the kinematic level. The advantage in doing so is the possibility of exploiting process redundancy for optimization purposes. Another decisive convenience of kinemati-

cally designing manipulator motion resides in the simple interfacing possibilities with the manipulator controller. The following sections will explain the steps which a scan trajectory undergoes to ensure that the corresponding motion of the manipulator meets the three above-mentioned constraints.

7.3 Time Efficient Motion Planning

7.3.1 Configuration Planning

The scan trajectory may sufficiently indicate the sensor's pose but the configuration, i.e. the joint values by which the FIS should approach each of these poses, is still uncommitted. The reason for this is that due to the non-uniqueness of the mapping from the Cartesian to the joint space, the inverse kinematic of the FIS may yield up to eight configurations by which each viewpoint could be reached. Figure 7.2 shows a graph that enumerates all possible configurations of the FIS and how they materialize. The graph is intended to be read top-down, with each branch representing a complete FIS configuration.

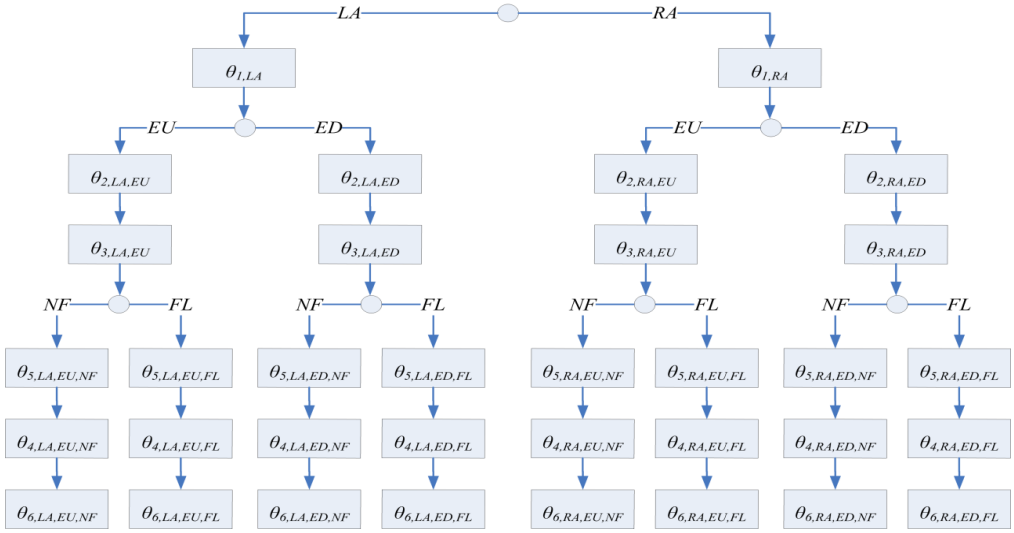


Figure 7.2: Graph representing the eight solution set of the inverse kinematic. Abbreviations are as follow: RA = Right Arm; LA = Left arm; EU = Elbow Up; ED = Elbow Down; NF = No Flip; FL = Flip

Although it is useful to have such great flexibility at one's disposal, great caution must also be exercised when configuring the posture by which the FIS should approach the viewpoints. Some of those postures may be invalid as they will lead to the FIS interfering with obstacles (e.g. surrounding components). For instance, figure 7.3 schematically depicts alternative configurations of the FIS for reaching

the same viewpoint. The "Elbow Up" configuration is invalid because the FIS interferes with the ceiling grid and should be therefore ignored during the configuration planning. Moreover, arbitrarily selecting FIS configurations along scan trajectories may result in an increased amount of wasteful configuration changes during the digitizing process. Configuration changes occur when one or more of the manipulator joints reaches its specific software limits. This forces the manipulator to reverse the course of the incriminated joints. As a result, the manipulator reaches the intended pose with a configuration that is different from its departure configuration.

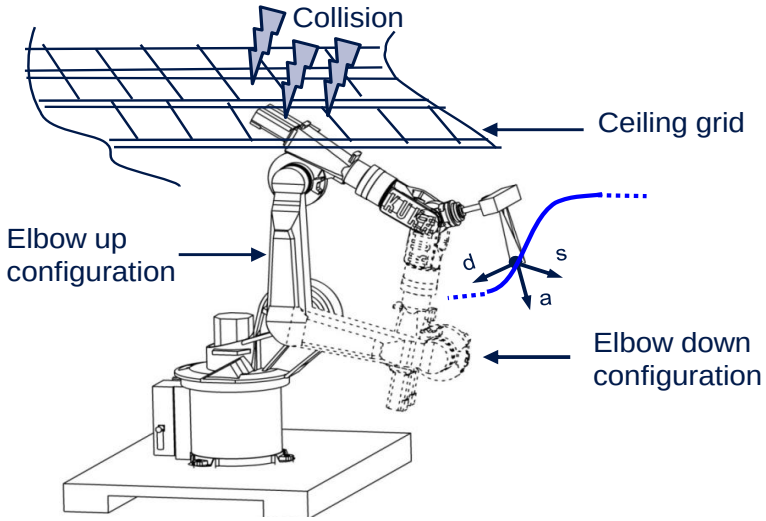


Figure 7.3: Different manipulator postures for the same viewpoint

Because such configuration changes divert a precious portion of the anyway tightly dimensioned cycle times, it is good practice to suppress them when possible (KONIETSCHKE 2007 p. 11). Of course, the best way to do so is to force the manipulator joints to operate in ranges distant from their specific limits. However, since scan trajectories are planned in the task space, keeping joints away from their working range's boundaries during the planning stage is scarcely possible. Therefore, a method aimed at minimizing the amount of configuration changes occurring in the scan trajectory was devised. The approach is based on the assumption that configuration changes start and end at exactly the same viewpoint on the scan trajectories. It is also assumed that the FIS conserves a unique and identical configuration between two viewpoints. As *TagPoints*—and, by extension viewpoints—are spatially close to each other, it is likely that these assumptions hold true.

7.3.2 Minimizing the Amount of Configuration Changes

As first step toward coming to grips with the issue of configuration changes, all the possible inverse kinematic solutions at each trajectory viewpoint are computed.

Next, all non-valid configurations are discarded. "Non-valid" means configurations that lead to interference between the FIS and its environment. Interference checks are performed by the collision engine that is integrated into the simulation module. The results of this pre-processing step are entered into a two-dimensional matrix, called *configuration matrix*. The matrix's columns represent the viewpoints, the rows the possible configuration postures, enumerated from one to eight. An occupied matrix entry (knot) signifies that the FIS configuration at the given viewpoint is valid. Figure 7.4 represents the configuration matrix of a scan trajectory comprising seven viewpoints. From this, it is obvious that more than one sequence of configurations exist to pass through all viewpoints. In such a case, randomly selecting the start configuration, for instance choosing C_1, is sub-optimal, because it will force the FIS to change its configuration three times as follows: $C_1 \rightarrow C_6 \rightarrow C_3 \rightarrow C_1$. In contrast, starting with the more favorable configuration C_2 will lead the FIS from the start to the end viewpoint through the configuration sequence $C_2 \rightarrow C_6$, which comprises only one configuration change.

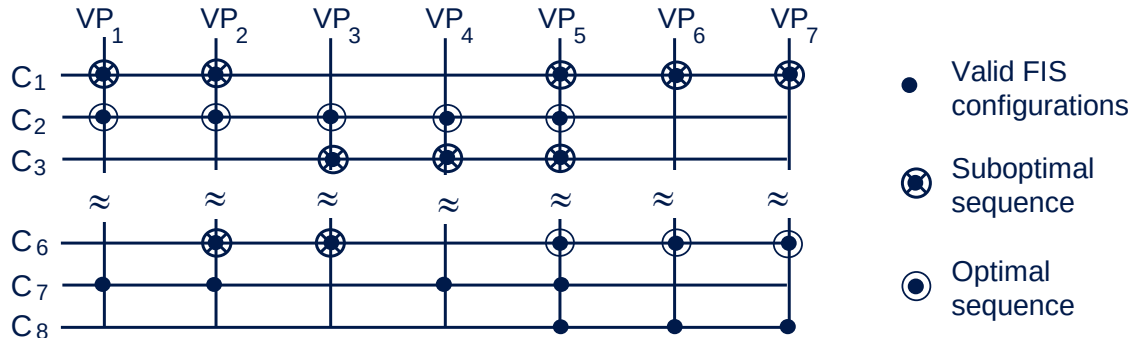


Figure 7.4: Configuration matrix, featuring an optimal and a sub-optimal sequences of configurations for traversing the same of viewpoint.

To minimize the configuration changes, the task configuration matrix is regarded as an undirected graph whose nodes are the knots and the (undirected) edges are the connection segments between neighboring knots. Nodes without direct successors or predecessors are discarded from the graph, since they would trap the search algorithm in a deadlock. Vertical edges are biased with a weighting coefficient proportional to the distance that the FIS's TCP would be required to travel during the configuration change. In this way, configurations which provoking huge detours from the task trajectory—thus unnecessarily extending the process cycle time—are penalized. In contrast, horizontal edges are not weighted since they do not initiate configuration changes. As the number of viewpoints is expected to be far higher than the seven taken in the example above, the search graph is reshuffled to save computation time. The reshuffling is carried out by merging graph nodes that can be horizontally traversed at a stretch to "expanded nodes." The left side of figure 7.5

depicts the new topology of the graph of figure 7.4 after being reshuffled. On the right, the graph is depicted in the usual, more easily understood, tree-fashion. The edges' weighting coefficients have been omitted for the sake of clarity.

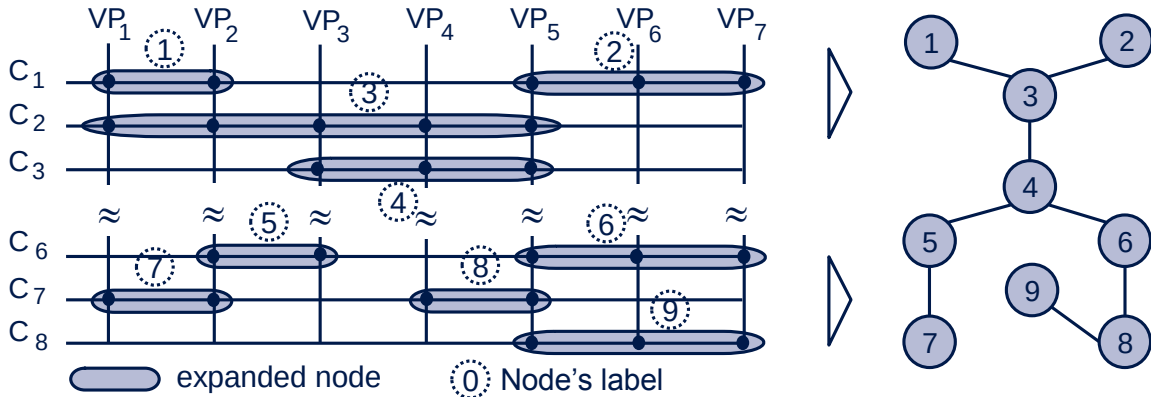


Figure 7.5: Reshuffled graph expressed in matrix representation (left) and in the labeled tree representation (right)

Based on this graph, the problem of finding the sequences minimizing the configuration changes is recast as a path-search problem, i.e. that of finding graph nodes which minimize the "traversing cost" for a non-backtracking path from an entry to an "exit" node. The "cost" of traversing the graph is thought to be the sum of the weights assigned to the path constituent edges. Referring to the graph from figure 7.5 (left), the entry nodes would be nodes 1, 3 and 7, whereas exit nodes are nodes 2, 6 and 9. The search problem is solved using a *Djistra algorithm*. For reasons of brevity further implementation details on the algorithm are not given here, brevity since the algorithm closely matches the one introduced by DIJKSTRA (1959).

To improve the performance of the path optimization procedure, expanded nodes that do not comprise a minimum travel distance are removed from the graph. The rationale behind this is that each time that a configuration change is performed the FIS has to be decelerated from its current speed, withdrawn from the scan trajectory and accelerated after having rejoined the scan trajectory. For short scan trajectory segments, this leads to the degeneration of the velocity profile from a trapezoid to a triangle, as shown in figure 7.6. The consequence of this is that the *VAConstr* is transgressed at those trajectory segments. To anticipate such deficiencies, a minimum size is imposed on the "expanded nodes." All those which include fewer than three viewpoints are discarded from the graph. Applied to the illustrative example above, it means that nodes 1, 5, 7, and 8 are no longer part of the graph. Since this "minimum size constraint" may drastically reduce the connectivity of the graph, it is therefore only applied when the graph topology renders more than one traversing sequence though the viewpoint possible.The method terminates by augmenting the data structure in which the scan trajectory is stored with additional information that

fixes the configuration to be adopted by the FIS at each viewpoint.

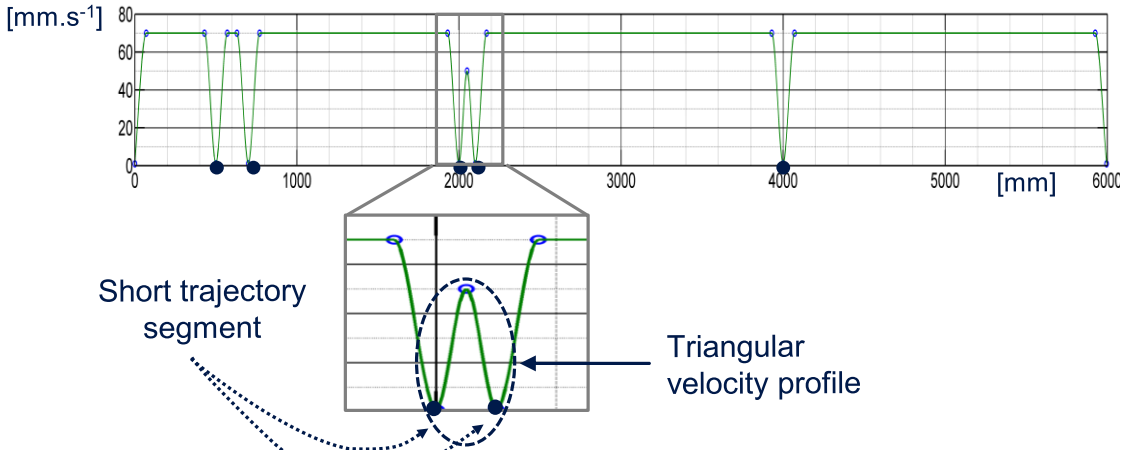


Figure 7.6: Degeneration of the velocity to a triangular form

7.4 Ensuring Collision-Free Motion of the FIS

7.4.1 Preliminaries

The motion executed by the FIS to digitize a WP can be broken down into transfer and constrained motion. During transfer motion, the FIS is commanded to move as quickly as possible from one configuration to the next, irrespective of the trajectory followed by its TCP. In contrast, constrained motion refers to those movements where the TCP is constrained to follow predefined scan trajectories. Whatever the motion type is, whenever the FIS moves, it must avoid contact with surrounding workcell components. Figure 7.7 illustrates the trajectory described by the FIS's TCP during a constrained linear motion interrupted by an unconstrained transfer motion from the "Flip" to the "No Flip" configuration.

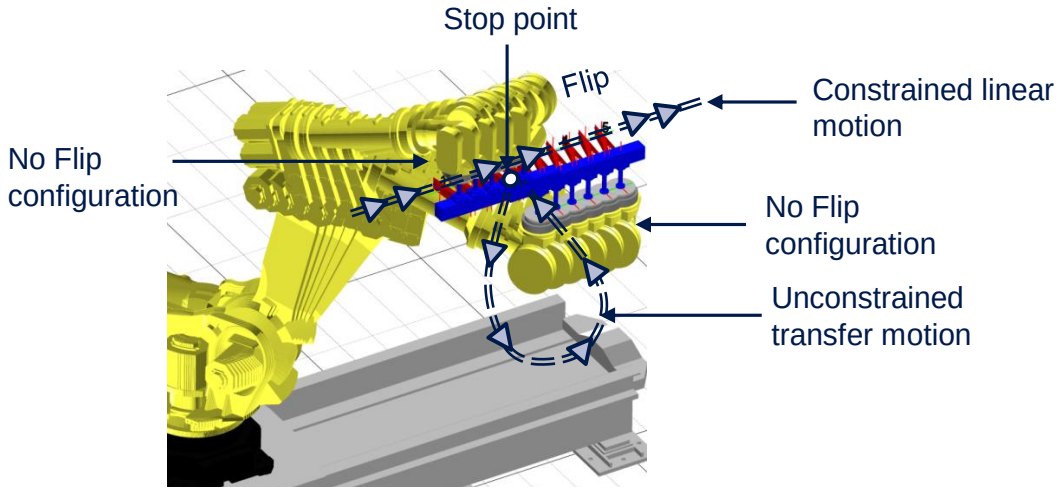


Figure 7.7: Description of the notion of transfer and constrained motion

7.4.2 Planning for Collision-Free Transfer Motions

An important aspect in the programming methodology is the generation of trajectories which allow the FIS to safely approach and withdraw from the WP. These kinds of motions are necessary at the beginning and at the end of the digitization process, where the robot has to be brought from its home or rest position to the WP and back. They are also needed each time the robot needs to perform a configuration change or when it moves from one task to the next without going through its rest posture. Transfer motions are characterized by the fact that the trajectory followed by the TCP is not subjected to any constraints. Therefore, it makes sense to shift the planning activities for transfer motion from the task space to the configuration space where the manipulator's posture is defined by its joint variables.

Combinatorial versus sampling motion planning

Two main philosophies have emerged in the course of the years to address the issues of motion planning in configuration space. On the one hand, there are combinatorial planning methods, which plan the manipulator motion based on an exact representation of the configuration space[22]. This leads to complete planning algorithms that are guaranteed to return a solution whenever one exists, or correctly report failure otherwise. However, their major drawbacks include difficulty of implementation and their prohibitively long running time. LAVALLE (2006 p. 80) even labeled them as unappealing for "industrial-grade" problems. As an answer to these limitations, sampling-based motion planers were developed around the mid-1990s. These operate by replacing the costly computation of the joint space by a collision test on every randomly picked robot configuration sample. The path is obtained by conducting discrete searches on these samples. The completeness of the algorithm is relaxed for the benefit of probabilistic completeness. Framed differently, this means that sampling-based motion planners are guaranteed to find a path provided that one exists. In doing so, the probability of finding such a path quickly converges to one, as the number of sampled states increases (KAVRAKI ET AL. 1998). They only downside of sampling-based methods is that their probabilistic nature leads to large standard deviations in the planning time (OJDANIC 2009 p. 69). Since the FIS is to be programmed off-line, this argument does not carry much weight. For an extensive treatment on the theory behind sampling motion planning, readers are referred to LATOMBE (1991) and LAVALLE (2006).

Probabilistic roadmap kernel

Because of the large effort necessitated for the implementation of a sampled motion planner, it appears that fitting an existing sampling planner into the motion planning

[22] Configuration space refers to the space of all possible placements of a robot.

module instead of developing a totally new one is a more productive approach. In this account, the open-source sampling motion planner named MPK[23] and jointly designed at the Simon Fraser and Stanford Universities was plugged into the motion planning module. This planner implements a *Probabilistic Roadmap* (PRM) planning strategy, which is the current leading sampling-based motion planning technique (GERAERTS & OVERMARS 2006). PRM-based planning approaches operate by randomly sampling the configuration space and retaining collision-free samples as milestones. Local search heuristics, such as the *k-nearest neighbor*, then try to locally connect milestones with straight-line segments. If the search heuristic fails to do this, further milestones are added and an attempt to connect the milestones is relaunched. Finally, collision-free connections are retained as feasible paths. This results in an undirected graph called *probabilistic roadmap*, whose nodes are the sampled milestones and the edges are the local feasible paths (see figure 7.8).

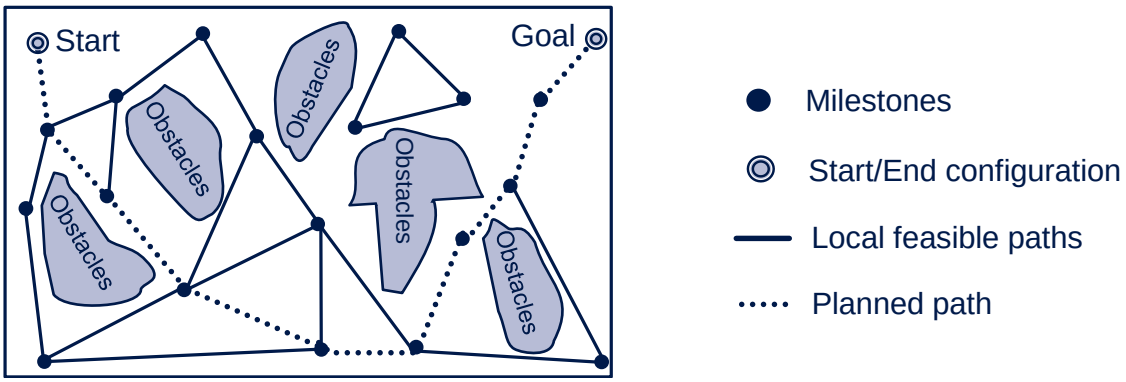


Figure 7.8: Probabilistic roa map with a planned connection path between start and goal configurations (dashed line)

Against this background, the FIS transfer motions are automatically generated by starting a query to the MPK with one or several start and goal configurations between which a collision-free motion is sought. For instance, the start configuration for the transfer motion that leads the FIS to approach the WP is its departure posture, e.g. its rest posture. The goal configuration is the configuration to be taken by the FIS at the first viewpoint on the scan trajectory. The same procedure is adopted for the transfer motions that withdraw the FIS from the WP, though in the reverse order. Automatic configuration changes during constrained motion—i.e. during execution of a scan trajectory—in contrast are planned by passing the FIS posture immediately before and after the stop point. In order to account for modeling uncertainties in the virtual workcell model, the sampling algorithm is

[23] MPK stands for "Motion Planning Kernel". A version of this software can be downloaded at: http://ramp.ensc.sfu.ca/mpk/downloads.html. The Stanford version, which is slightly different, is labeled the "Motion Planning Kit" and can be downloaded at: http://ai.stanford.edu/~mitul/mpk/index.html.

biased with a safety margin to control how close the sampled configuration should be from obstacles.

The output of each query to the PRM is an ordered sequence of 6-tuple joint values which correspond to the joint values that the FIS has to follow in order to safely achieve the transfer motion. As no process constraints are to be applied to transfer motion movements, there is no need to design a velocity profile for them. Hence, the freedom is given to the manipulator control unit to choose the smoothest and fastest way to implement the transfer motion at execution time.

7.4.3 Planning for Collision-Free Task Constrained Motions

Some segments of the scan trajectory may cause the manipulator structure to interfere with surrounding workcell components. Because of its probabilistic nature, using the MPK to generate collision-free constrained motion is not an option (SENFT & HIRZINGER 1995). The likelihood that a PRM planner (randomly) synthesizes a configuration sequence which leads the TCP of an IR along a predefined trajectory is close to zero percent, not to say impossible (STILMAN 2010). For this reason, the effective integration of task constraints into sampling-based motion planning frameworks is still an unresolved matter (SHKOLNIK 2010 p. 102). The approach chosen in this module is to finely pre-sample the scan trajectory in equally spaced poses and check the FIS posture at each of them for collision. To balance the trade-off between accuracy and efficiency of this procedure, a discretization step $\Delta s = 10\,mm$ with respect to the arc length s is applied. Since the *configuration matrix* only encodes the configuration of the FIS at the viewpoints, a sampled pose that lies between two consecutive viewpoints is attributed the configuration of the FIS at the preceding viewpoint.

Each time that the collision engine reports a positive hit, a procedure is triggered that reshapes the portion of the scan trajectory on which the blameworthy sensor pose lies. The procedure operates by first tracing back the three consecutive viewpoints that border the faulty pose on each side. Next, the *scannability frustums* from which these viewpoints have been originated are retrieved. From those *scannability frustums*, new viewpoints are selected and substituted for the previous ones. To increase the probability of capturing new viewpoints that may remove the collision handicap, the viewpoints are chosen randomly. After that, the viewpoints are interpolated to reconstitute a locally reshaped scan trajectory, which is once again tested for collision. If a collision handicap still remains, the entire procedure is repeated with a slight modification. The portion of the trajectory to be reshaped is extended to five additional viewpoints on each side of the faulty pose. Should the collision handicap still not be eliminated after five iterations, then this is an indica-

tion that the FIS cannot steer these trajectory in the current workcell arrangement. Accordingly, the motion planning module prompts the user to a) relocate the part according to the *scannability map* and/or b) get rid of dispensable components which currently obstruct the FIS workcell. It must be pointed out that relocating the WP leaves the scan trajectory intact so that the programming task can be resumed directly at the stage of motion planning.

7.5 Singularity-Free Motion Planning

7.5.1 Preliminaries

While tracking scan trajectories, the FIS may move close to, or even worse drive through kinematic singularities. In the neighborhood of singularities, small TCP Cartesian velocities unleash large joint efforts (SCHREIBER 2005 p. 37, SPONG ET AL. 2006 p. 142). Situations in which these excessive joint efforts are well beyond the capabilities of the manipulator's actuators, leading the controller to issue an emergency stop, are not uncommon (TSAI 1986). It is therefore imperative to clear the scan trajectories of singular points in order not to jeopardize their feasibility.

Manipulator singularities can be classified into workspace-interior and workspace-boundary singularities (SCIAVICCO & SICILIANO 2005 p. 90). The latter type of singularities occur when the manipulator arm is fully outstretched or folded back on itself so that the TCP is at or very near to the workspace boundary (HOLLERBACH 1989 p. 389). From this, it is clear that all kind of manipulators possess workspace-boundary singularities, regardless of their kinematic structures. However, these do not represent a true drawback since they can be avoided simply by preventing the manipulator from moving toward the workspace barrier, which is uninteresting anyway.

In contrast, the more challenging workspace-interior singularities can arise anywhere, as soon as two joint axes lines up (SICILIANO & KHATIB 2008 p. 252). Figure 7.4 displays a singular FIS posture characterized by the fact that axis 4 and 6 coincide. According to LLOYD & HAYWARD (1993), one of the three following strategies can be taken to escape singularities: a) deviating from the curve while keeping the time law unchanged; b) accurately tracking the path while relaxing the given time law; c) performing some combination of the both. As slowing down the FIS during the digitizing process is inconsistent with the *VelConstr*, the alternatives b) and c) can be disqualified a priori. In evidence of the fact that deviations which do not result in shifting the sensor's FoV away from the targeted AoI, will not penalize the digitizing process, alternative a) seems to be the best option.

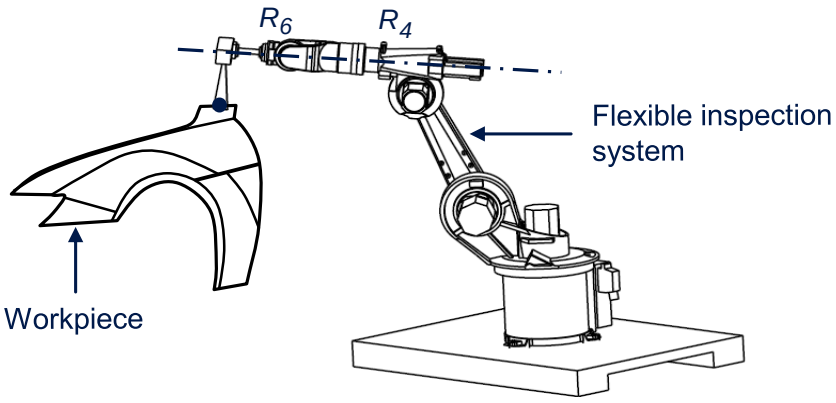


Figure 7.9: Singular configuration of the FIS

7.5.2 Overcoming Singularities along the Scan Trajectory

7.5.2.1 The Damped Least-Squares Methods

Damped Least-Squares (DLS) designates a kinematic strategy to control the motion of a manipulator in the vicinity of kinematic singularities that can be traced back to the work of WAMPLER (1986) and NAKAMURA & HANAFUSA (1986). It is based on the manipulator's Jacobian J, which relates differential changes in joint coordinates $\Delta\Theta = [\delta\theta_1\ \delta\theta_2\ \delta\theta_3\ \delta\theta_4\ \delta\theta_5\ \delta\theta_6]$ to differential changes in Cartesian coordinates $\Delta p = [\delta x, \delta y, \delta z, \delta\alpha, \delta\beta, \delta\gamma]$ as follows:

$$\Delta p = J\Delta\Theta \tag{7.1}$$

The tenet of the DLS is to "damp" differential joint motion $\|\Delta\Theta\|^2$, at the expense of increasing differential end-effector path-tracking errors $\|J\Delta\Theta - \Delta p\|^2$. In other words, this means that the deviation between the desired robot motion Δp and the reconstructed singularity-free real motion $J\Delta\Theta$ should be as small as possible. Simultaneously, the joint differential motion $\Delta\Theta$ has to be kept as low as possible. WAMPLER (1986) and NAKAMURA & HANAFUSA (1986) recast this problem as a Levenberg-Marquardt minimization problem[24] and express it as follows:

$$\min_{\Delta\theta}\{\ \|J\Delta\Theta - \Delta p\|^2 + \lambda^2\ \|\Delta\Theta\|^2\} \tag{7.2}$$

where λ is a strictly positive damping factor to be appropriately determined. From LEVENBERG (1944) the solution for equation (7.2) can now be written as

$$\Delta\Theta = (J^T J + \lambda^2 I)^{-1} J^T \Delta p \tag{7.3}$$

where J^T designates the transpose of J and I is the identity matrix.

[24] The Levenberg-Marquardt algorithm iteratively locates the minimum of a multivariate function that is expressed as the sum of squares of nonlinear real-valued functions (LEVENBERG 1944).

At singular points, J loses rank and becomes non-invertible (SCHREIBER ET AL. 1999). To avoid this, J is substituted by a singularity robust approximation in term of the singular value decomposition (SVD), as shown in the equation below:

$$J = U\Sigma V^T = \sum_{i=1}^{6} \sigma_i u_i v_i^T \tag{7.4}$$

where U, V are unitary square matrices and $\Sigma = diag\,[\sigma_1 \ldots \sigma_6]$ is a diagonal square matrix whose main diagonal is filled with the singular values of J. When one or more of these singular values are close to zero, this is an indication that the manipulator is approaching a singular point. Substituting equation (7.4) into equation (7.3) leads to

$$\Delta\Theta = \sum_{i=1}^{6} \frac{\sigma_i}{\sigma_i^2 + \lambda^2} v_i u_i^T \Delta p \tag{7.5}$$

From equation (7.5), it is clear that using a constant value for λ will damp the FIS over the entire workspace, e.g. at trajectory segments which are far from singularities. To avoid this, the adaptive strategy introduced by CHIAVERINI ET AL. (1994), which consists of adjusting λ as a function of its closeness to the singularity is used, as shown in the equation below.

$$\lambda^2 = \begin{cases} 0 & when \quad \hat{\sigma} \gg \tau \\ \left(1 - \left(\frac{\hat{\sigma}}{\tau}\right)^2\right) \cdot \lambda_{max}^2 & otherwise \end{cases} \tag{7.6}$$

In this equation, τ denotes a factor defining the size of the singular region, $\hat{\sigma}$ the smallest singular value of the Jacobian while λ_{max} denotes the maximal damping value that can be used to adjust the solution of equation (7.5) to the specific needs of the robotic application under investigation. Simulation experiments carried out by CHIAVERINI ET AL. (1994) have shown that delimiting the singular region by $\tau = 0.04$ is a good compromise with which to start. It should be noted that, when λ is set to zero, equation (7.3) is equivalent to a pure least-square approximation of the inverse kinematic solution.

7.5.2.2 Singularity Avoidance Strategy

To clear the scan trajectories from kinematic singularities, an algorithm that draws on the DLS is implemented within the motion planning module. This algorithm operates as follows: first, the scan trajectory is finely sampled in a sequence of evenly spaced sensor poses p_k. Next, using equation (7.5), the differential changes between the sampled pose Δp_k are converted to equivalent manipulator joint incre-

ments. From this, the joint values corresponding to each sampled pose p_k are retrieved by iteratively adding the motion increment to the previous joint values as follows: $\Theta_{k+1} = \Theta_k + \Delta\Theta_k$. It should be noted that for this strategy to work, the first pose p_0 along the scan trajectory has to be non-singular. As scan trajectories are intended to be approached by the FIS with a transfer motion that are expressed in the joint space (see section 7.4.2), this assumption holds true anyway[25].

It should be stressed that the DLS is expected to alter the orientation of the FIS's TCP during the singularity damping process (SCHREIBER ET AL. 1999). As the constraints *VAConstr*, *CleaConstr* and *VisConstr* are sensitive to changes of the sensor orientation, these are to be avoided. For this reason, the changes in the sensor orientation induced by the DLS have to be reversed. To do so, the vectors of joint values $[\Theta_k]_{k=1\ldots n}$ are fed to the FIS forward kinematic to convert the singularity-free scan trajectory from the joint into the task space. Then, for each sample pose $\check{p}_k\left(\check{X}_{FP}\ \check{Y}_{FP}\ \check{Z}_{FP}\ \check{\gamma}_{FP}\ \check{\beta}_{FP}\ \check{\alpha}_{FP}\right)_k$ obtained from a 6-tuple Θ_k, the orientation parameters $\left(\check{\gamma}_{FP}\ \check{\beta}_{FP}\ \check{\alpha}_{FP}\right)_k$ are substituted by those that were in the scan trajectory before it underwent the singularity damping process, i.e. $(\hat{\gamma}_{FP}\ \hat{\beta}_{FP}\ \hat{\alpha}_{FP})$. In this way, only the translational scan trajectory is used to maneuver the FIS around the singularities.

One major drawback of the DLS method lies in the fact that it can suffer from deficiencies in terms of bad tracking accuracy imputable to the improper determination of λ_{max} or respectively λ. To achieve the best possible compromise between feasibility (limited joint effort of the FIS) and the accuracy (small deviation from the desired trajectory) of the DLS method, λ_{max} is tweaked until the tracking accuracy drops below a threshold value S_{max}. In doing so, the first estimate of λ_{max} is set to half of the largest singular value. S_{max} represents the radius of a sphere centered at the sensor focal point whose intersection surface with the laser plane does not cross the FoV's boundaries (see figure 7.10).

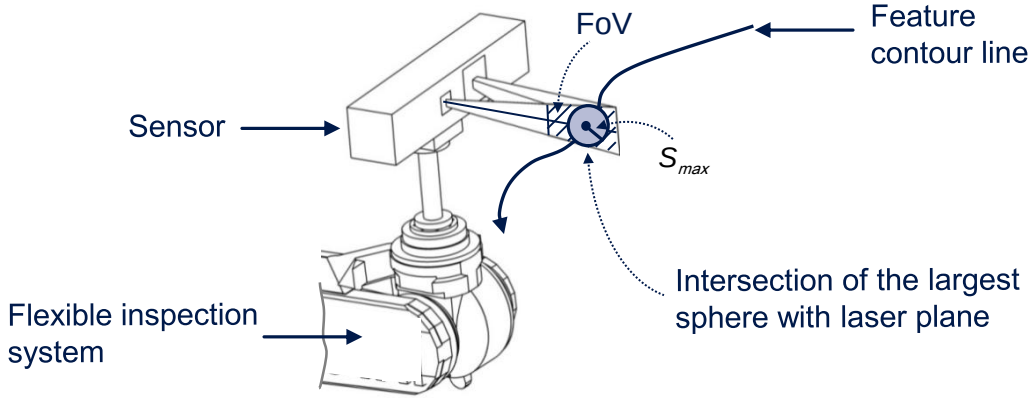


Figure 7.10: Principle of the threshold value S_{max}

[25] Singularities issues arise only when manipulators are maneuvered or controlled in the task space.

The spirit and purpose of this threshold value is to ensure that the feature line intended to be digitized still falls into the sensor's FoV despite the deviation. Otherwise the FIS will successfully drive around singularities but may miss a segment of the feature line it should have captured.

Figure 7.11 shows how the singularity avoidance method acts on a singularity-prone scan trajectory. The upper subframe of this figure displays the translational scan trajectory before and after the singularity avoidance strategy was applied on it.

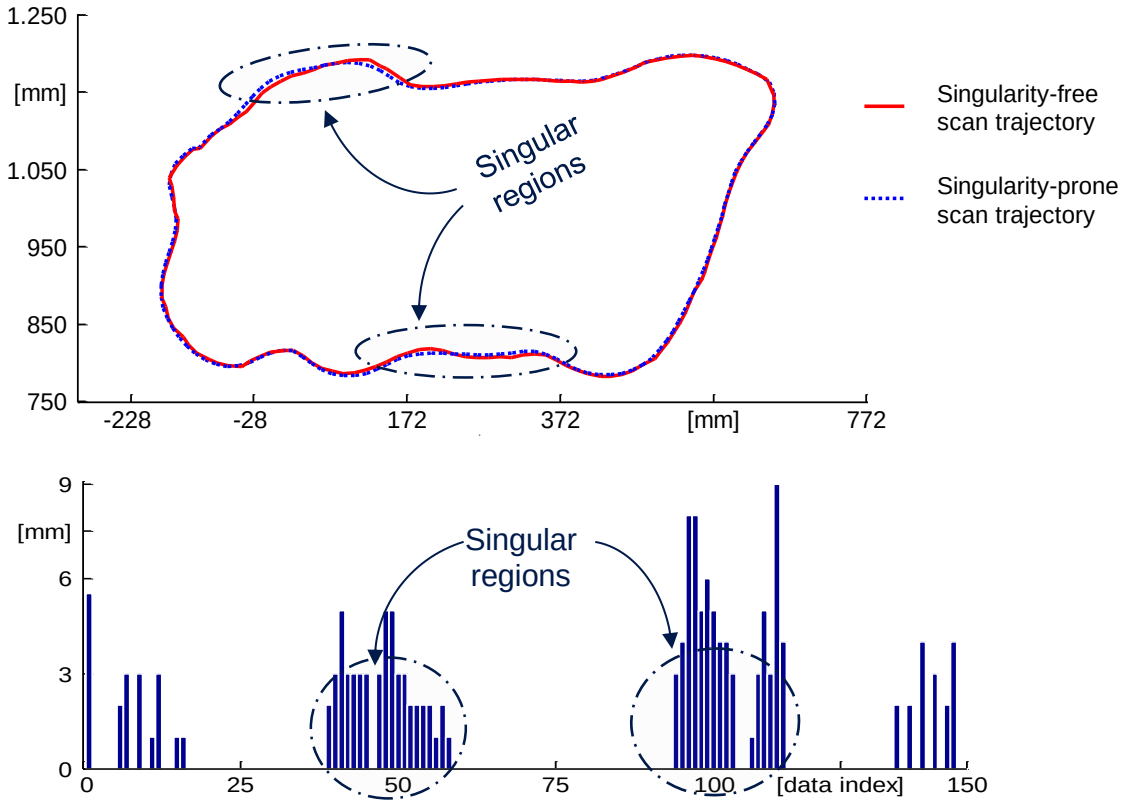


Figure 7.11: Effect of the DLS on scan trajectories a)translational scan trajectories; b) Deviaition between the the singularity-free and the singularity-prone scan trajectory

A close observation of both curves reveals two clear singular regions. In these regions, one can identify how the calculated singularity-free trajectory deviates from the singularity-prone trajectory and then crosses it to drive around pending singularity obstacles. The bar graph in the lower subframe depicts the absolute value of the deviation between the singularity-free and the singularity-prone trajectories at each interpolation point. There, singular regions are identifiable by the intensity and the height of the deviation bars. Within singular regions, crossing points between both trajectories are spotted where the deviation bar vanishes. In addition, this chart also shows that the error induced by the DLS in the singularity-free scan trajectory is constantly below 10 mm, which happens to be the radius of

the tolerance sphere S_{max} considered during the simulation. As this error is expected to leave the altered scan trajectory within the FoV of the sensor, it can be deduced that the tracking accuracy of the singularity avoidance strategy does not jeopardize the realization of the digitization task.

7.6 Considering the Manipulator's Dynamic Limitations

So far, the scan trajectories and FIS motion have been planned without explicitly considering the manipulator dynamics. Since manipulators' actuators likewise possess limited capabilities, the magnitude of the torque effort necessary for the FIS to steer a given trajectory should not exceed these limits. In order to incorporate torque control at the phase of robot motion planning, algorithms based on the manipulator's dynamic model are required (FAIZ & AGRAWAL 2000). One then speaks of dynamic motion control. Implementing an accurate dynamic model of an IR is however not a trivial undertaking, especially for manipulators. This can be attributed on the one hand to the highly nonlinear and coupled nature of manipulator dynamics, and on the other hand to model parameters which are difficult to determine, e.g. arm compliance, motor torque ripple, coulomb friction. SHEN (2005) argued that trajectory planning methods based on the manipulator dynamic model will often end up being sub-optimal due to the poor model fidelity.

Even if the dynamic models were accurate enough, torque-based motion planning would still not be practical. The reason for this is that their implementation necessitates the substitution of the vendor specific low-level joint servos on the manipulator controller with custom control loops (ANTONELLI ET AL. 2003). However, low-level servos are encapsulated in a software layer that the robot manufacturers are always eager to keep the consumer from tempering with (BROGÅRDH 2009, SOMMERKORN ET AL. 2010). They worry about unsafe behaviors of their products that could lead to human casualties and damage to their reputation. Therefore, it seems more productive to concentrate on motion planning approaches that do not necessitate comprehensive dynamic models. Fortunately, as demonstrated in sections 7.3, 7.4, 7.5, robot motion can not only be planed on the basis of dynamic models, but also based on kinematic models. One then speaks of kinematic motion control. Nonetheless, kinematics models, just like dynamic models, must also be a faithful representation of their physical counterparts. However, as the magnitude of error encountered in such models is likely to be less than those of dynamic models, they are more trustworthier for motion planning purposes.

In the framework of kinematic motion control, acceleration constraints play the counterpart of the joint torque limits in dynamic motion control. According to ANTONELLI (2003) and VAN DIJK ET AL. (2007), in planning a motion pattern that

does not violate the manipulator's physical limits, it is sufficient to control the acceleration magnitude invoked in the actuators. An indirect method of achieving this is to put some restriction on the curvature of the trajectory to be tracked by the TCP (MATTMÜLLER & GISLER 2007). Then the smoother the trajectory, the more likely it is to require reasonable acceleration, and hence, realizable torque effort in the manipulators' actuators (CONSTANTINESCU & CROFT 2000). From this one may infer that, even though the manipulator dynamics have not been explicitly modeled in the motion planning module, the scan trajectory generation process implicitly accounts for them. To prove this, it is recalled that the scan trajectories have already undergone two smoothing processes. Additionally, the torque demand in the FIS joints is further alleviated by the bell shape of the velocity profile.

Last but not least, it should be pointed out that during the digitization task, the manipulator is not operated at the limits of its physical performance. FISs are usually operated at a scan speed of maximal 80 $mm.s^{-1}$ (PERCEPTRON 2011), which is far below the manipulator attainable maximal speed, e.g. 2500 $mm.s^{-1}$ (FRANKE & DOBROSCHKE 2010). The reasons for this can be found in the limited data transfer rate between the FIS components as well as the increased vulnerability of the manipulator's mechanical structure toward vibrations at high operating speed. In addition, running the FIS at high speed can only be achieved at the expense of the range image density, which is supposed to be tight. On that score, the FIS is likely to stay within its physical bounds, except at singular points, where the efforts of the actuators surge anomalously. However, in light of the singularity avoidance method implemented in the previous section, the issue of handling excessive torque demands in the manipulator actuator at singular points is resolved.

7.7 Data Conditioning

Although some advanced robot controllers already implement splines interpolation techniques (e.g. KUKA 2011, REIS 2011, MOTORMAN 2001), there are still many controllers that do not (e.g. ABB 2011, FANUC 2011, STAUBLI 2011,). To account for this, task-constrained scan trajectories are not transmitted in the spline fashion. Instead, each of them is sampled using a fine, regular discretization scheme. A "LIN" instruction is augmented to each discrete point to ensure that the FIS execute straight-line motion between them and therefore closely follow the shape of the original scan trajectory. Transfer motion trajectories do not undergo the same sampling procedure as they do not need to. Rather, they are directly logged in the data structure and labeled with the command "PTP". This makes it clear to the controller that transfer motions are to be performed in joint interpolated fashion, e.g. the manipulator has the freedom to choose the fastest way to implement the movement.

7.8 Visual Servoing System

7.8.1 Closing the Gap between the Virtual and Real Workcells

As previously emphasized, the use of an off-line programming system requires the virtual workcell and its physical replicate to match closely. There are several obstacles which prevent this in practice to happen. Chief among these are the misplacement of components in the physical workcell and the use of theoretical models in the virtual workcell (BLEY ET AL. 2001). Idealized kinematic parameters utilized in the manipulator control unit and the temperature drift encountered in its mechanical structure may also share the blame for these discrepancies (HEISEL ET AL. 1995). According to ROOS ET AL. (1997) such flaws will lead to poor tracking performance of IRs during the execution of off-line generated trajectories. In the context of FISs, such poor tracking performance may result in the defocusing of some AoI on the WP, which may end up not being captured by the FIS.

To ensure that the real and virtual worlds match closely, time-consuming manual calibration procedures are usually applied in practice (NOF 1999 pp. 798–798). Expensive clamping or fixture devices (DEACON 1999) or optical calibration techniques are sometimes also deployed to restrict the DoF between the workcell items. Falling back on such conventional means is not an option in the context of this research, as they annihilate the time- and cost-saving currently pursued to make the deployment of FISs affordable for SMEs. In order not to jeopardize those incentives, some degree of adaptiveness has to be introduced into the programming methodology.

7.8.2 Conceptual Approach

A visual servoing approach may be a promising approach to bridging the modeling accuracy gap between the virtual and real worlds (MILBERG ET AL. 1997). Visual servoing refers to control techniques that rely on sensor feedback from optical sensor systems to achieve tracking or regulation objectives (HU 2007). Depending on the nature of the controlled variable, visual servoing methods can be classified into two groups (BACHILLER ET AL. 2007). One group comprises image-based methods where the control cues are computed in the 2D-image space in terms of image pixel errors; the other group is constituted of position-based techniques where the control variables are based on reconstructed Euclidian information. As LSSs directly report range information relative to the objects' surfaces, it goes without saying that with regard to FISs, position-based visual servoing is the efficacious way forward.

Accordingly, a position visual servo system was designed and integrated into the motion planning module. The goal of the regulation strategy is to keep the sensor's FoV immersed in the WP's surface. This is achieved by monitoring the sensor readings to determine if the WP is about to leave the FoV. If the sensor is becoming defocused, then appropriate corrective action is prompted. A prerequisite here is, of course, that, at the beginning of the digitizing process, the sensor's FoV is already plunging into the WP surface. If the deviations between the planned and the desired trajectory are so huge that this condition cannot be met, then the sensor will not report any depth information, causing the visual servoing feedback loop to fail.

Although the FIS collects 3D range data, it is not possible to meaningfully interpret all the three coordinate data for the servoing task. Only the coordinate data that capture the sensor's standoff distance to the WP, i.e. the z-coordinate, can be unambiguously interpreted. No valuable information can be robustly retrieved from the x- and y- coordinates on whether the FoV is drifting away from the WP. Therefore the servoing system endeavors to control only the sensor standoff distance to the WP. Recalling that each sensor reading consists of an array of range points, this standoff distance is computed by averaging the four most centrally located points within the array. To simplify the control law, the sensor readings processed in the visual servoing system are expressed in the laser source frame F_{LS} (see figure 7.12).

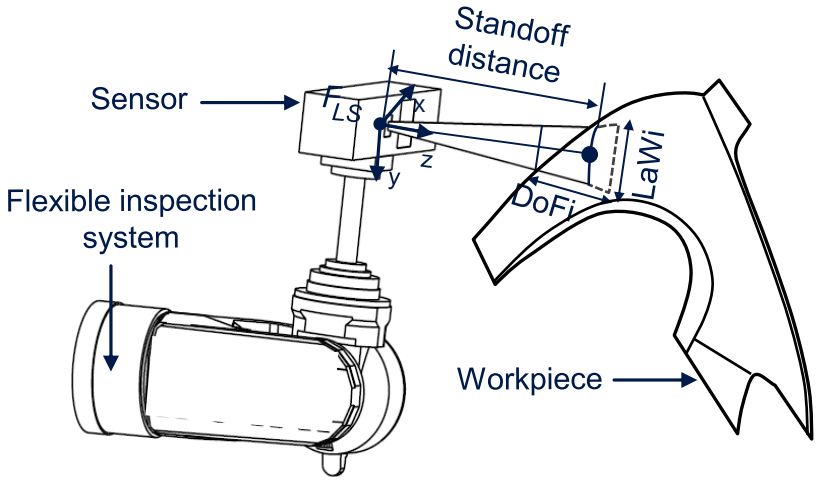


Figure 7.12: Principle of the standoff distance

7.8.3 Architecture of the Virtual Servoing System

Figure 7.13 depicts a block diagram showing how the visual servoing system interacts with the FIS. As can be seen, the visual servoing system sits in a cascade arrangement on the top of the built-in position control loop present in the manipulator control unit. The inputs to the visual servoing system are the sensor readings. The output is a path correction cue that is fed into a trajectory adaptation interface

with which IR control units are always equipped with. From these correction cues, the FIS is maneuvered in such a way that the WP is kept in the sensor's FoV.

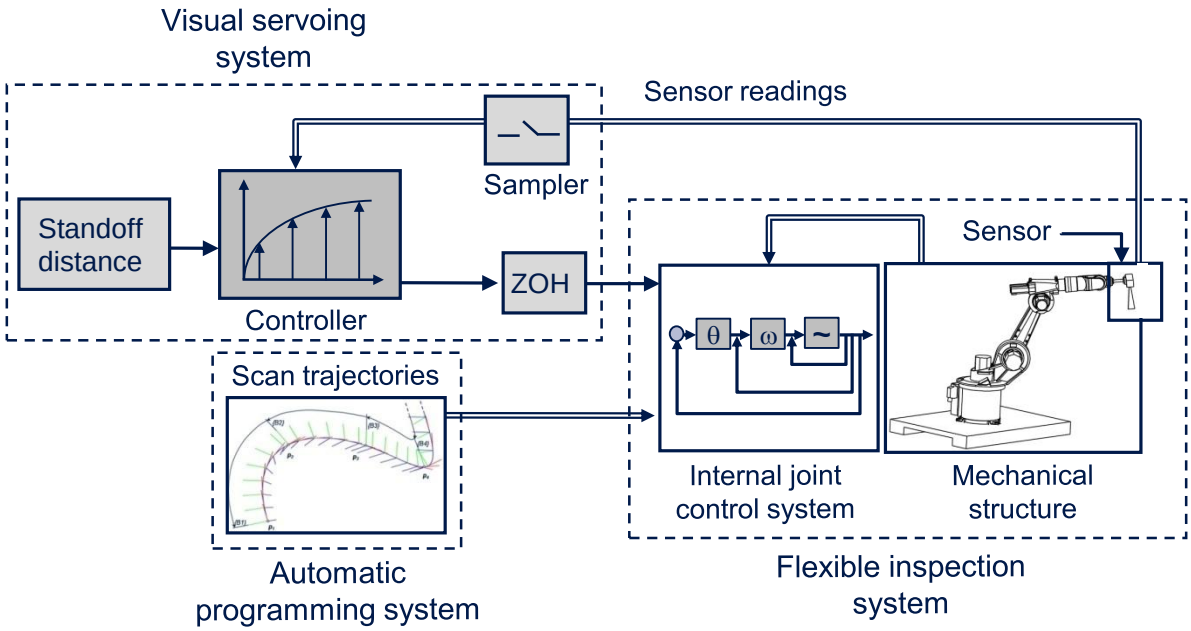


Figure 7.13: Block diagram of the visual servoing system

For safety reasons, servoing cues are paired with an additional command that forbid the FIS to adopt configurations other than the one specified in the portion of the scan trajectory being steered. This preempts unpredictable configuration changes that may lead to collision. Additionally, the servoing cues are saturated to the length of the FoV—i.e. the DoFi. This allows for ruling out large correction cues that may cause the FIS to crash against the WP. Also, the total deviation of the FIS from the programmed track is limited to be maximum twice the DoFi. The reason for this is that such a big deviation is beyond typical modeling errors encountered in offline programming. Such deviation figures are therefore interpreted as a strong indication that something fundamentally wrong occurred at the modeling stage.

7.8.4 Controller Design

The goal here is to design a simple and reliable controller that possesses good transient response, yields small steady state errors and does not induce additional vibration in the FIS's joints. As proportional-integral-derivative (PID) controllers are said to meet these requirements, the standoff controller is of the same type. Even though designing such controllers does not necessarily require any knowledge of the plant (BENNETT 1993 p. 48, SICILIANO & KHATIB 2008 p. 141), e.g. the manipulator dynamics, a model-based approach to design the control law was elaborated. Using a model approach bears the advantage of avoiding lengthy parameter tuning

loops until a well-pitched control system is set up that squares system responsiveness with system stability while minimizing steady-state error.

Black box system identification

Given the complexity and nonlinearity of manipulator dynamics properties, an rough dynamic model of the FIS is identified, rather than deriving one from laws of fundamental physical—*Euler-Lagrange* or *Newton-Euler formalisms*. To do so, the FIS is stimulated with a step signal. Prior to initiating the step command, the nominal acceleration of the FIS is set to 100% and its velocity is limited to 50 mm.s^{-1}, as this is the average velocity expected to be used in the experimental platform later on (see section 8.4). The sensor is positioned at a height of 170 mm above the WP surface and a step command is issued to move the FIS's TCP by 35 mm, up to a height of 205 mm. The sampling time of the sensor readings is 24 ms, which corresponds to the update rate of the communication channel (more on this in section 8.3). The open-loop response of the FIS to the step command is shown by the diagram on the left-hand side of figure 7.14.

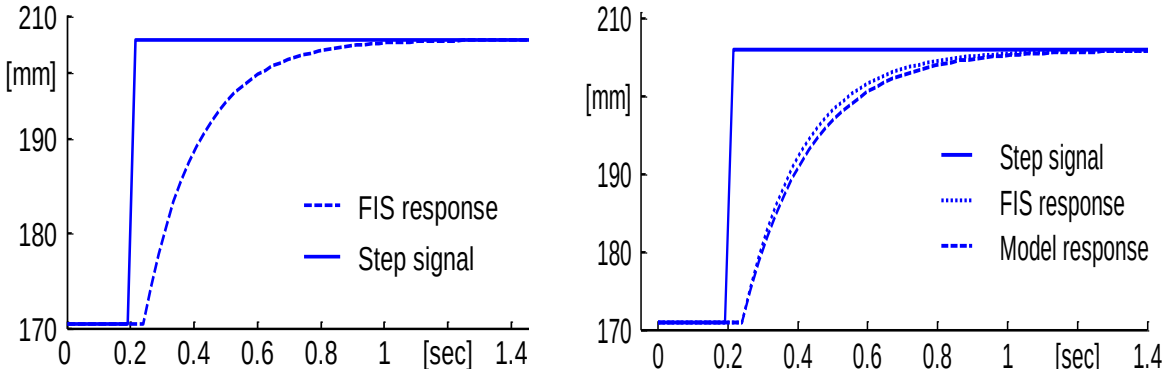


Figure 7.14: a) Step signal and time evolution of the FIS response; b) Comparison of the FIS step response and the response of the identified model

This response curve indicates that no physical motion occurs for 48 ms—twice the communication sample time—after the step command. Furthermore, this response curve exhibits no oscillation, no overshoot and a steady-state error approximately amounting 0 mm. Therefore it can be deduced that the FIS is BIBO[26] stabile (HELLERSTEIN 2004 p. 93). A first-order-plus-dead-time model (FOPDT) with two sample-time delays and a steady state gain set to one can be expected to approximately fit the step response (ROY & IQBAL 2005). From discrete control theory, it is known that a corresponding transfer motion takes the following form:

[26] A linear, time-invariant is said to be BIBO (Bounded Input Bounded Output) stable when its output to every bounded input is also bounded.

$$G = \frac{a}{1-(1-a)z^{-1}} \cdot \frac{1}{z^2} \quad with \quad a = \frac{1}{\frac{T}{T_s}+1} \tag{7.7}$$

where T represent the time constant and T_s the sampling time. Substituting the parameters in equation (7.7) for values yields,

$$G = \frac{0.324\, z^{-2}}{1-0.676z^{-1}} \tag{7.8}$$

The right-hand diagram of figure 7.14 features the open-loop step response of the real FIS and that of the identified model (i.e. the transfer function G). As can be seen the response curves matches fairly well, which is a strong indication that the identified model to some extend mirror the behavior of the physical FIS.

Model-based controller design

Based on the FIS model identified above, a PID control law is to be designed that characterizes the logic of how sensor data should influence the FIS motion with regard to keeping the part surface immersed in the sensor's FoV. The design objective is to choose the controller parameters (i.e. poles) to produce a fast-tracking and stable response of the FIS to the sensor readings. Equation (7.9) shows the ideal form of a PID transfer function in the z-transform domain

$$C = K_p \cdot (1 + \frac{z-1}{z} + \frac{z}{z-1}) \tag{7.9}$$

where K_p is the gain factor of the closed-loop system (uncontrolled FIS). The root locus diagram in terms of a varying K_p is shown on the left-hand side of figure 7.15.

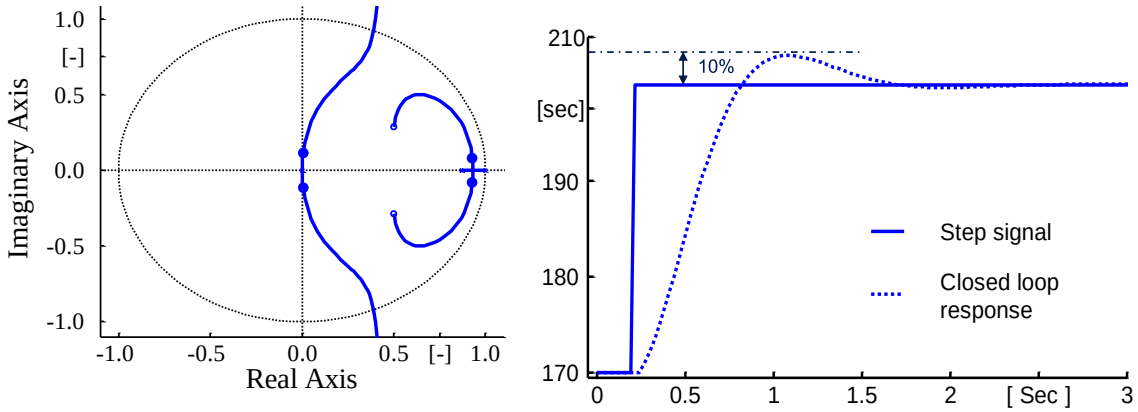


Figure 7.15: a) Root locus curve of the closed loop; b) Step response of the controlled FIS

Based on a root locus analysis, a closed-loop gain of $K_p = 0.092$ (damping ratio ξ =

0.69) was found to reproduce behavior of the FIS that is in a good agreement with the control objectives enumerated above. This perception is endorsed in the right-hand side of figure 7.15 showing the closed-loop response of the FIS's model when it is given the same step excitation input as in the model identification stage. As can be seen the time evolution of the system response displays a fast settling time (1.5 s), shows marginal overshoot (10%), and has virtually no steady-state error.

Test and conclusion

A stress test is conducted on the physical FIS to evaluate the performance of the stand-off controller. To do so, the controller inputs and outputs are rerouted from the FIS simulation model to its physical counterpart via a real-time data channel (see section 8.3). The test case consisted of commanding the FIS to track a scan trajectory while overlaying a disturbance signal on the sensor readings that are fed back to the controller. This disturbance signal is meant to emulate deviations that may arise between the desired and actual steered trajectories due to modeling inaccuracies. It is designed as a low frequency (0.25 Hz) sinusoidal function with peak amplitude set to 50 mm—approximately the DoFi of the LSS build used on the experimental platform.

The desired scan trajectory and the sensor readings recorded during the experiment are plotted on the right-hand side of figure 7.16. They show that the disturbance signal is successfully rejected by the standoff controller, as the maximum deviation of the FIS from the desired scan trajectory is less than 20 mm. In other words, the sensor is kept at a standoff distance, which maintains the sensor's FoV in focus. From this, it can be concluded that the PID parameter setting is in good agreement with the control objectives formulated above.

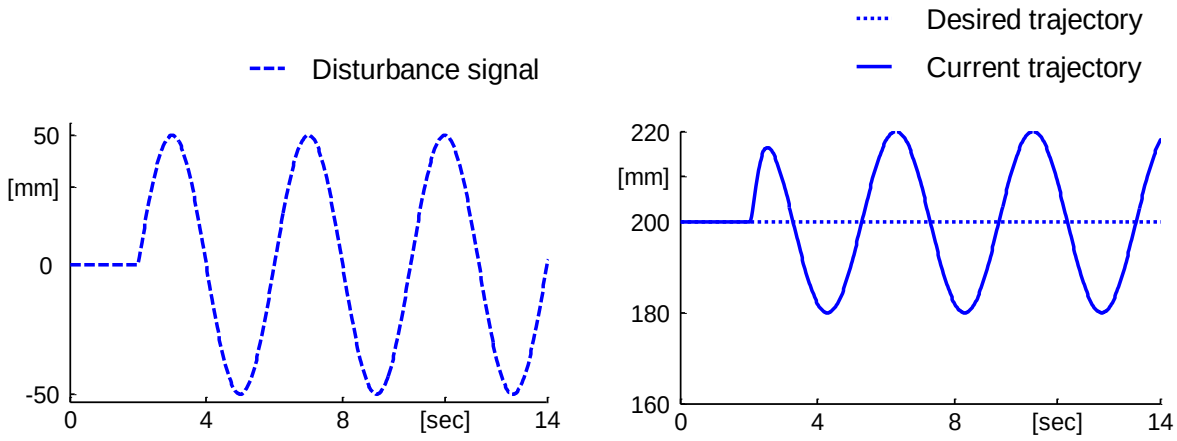


Figure 7.16: Disturbance signal applied to the FIS (left) and response of the controlled FIS (right)

8 Integration, Demonstration and Evaluation

8.1 Overview

This chapter describes the implementation details and the integration of the four modules described in the previous chapters into one homogeneous programming system. In addition, the communication architecture that interfaces the FIS components with the programming system is discussed. To furnish evidence of the methodology's achievement potentials, an industrial case study will be performed on an experimental setup. In addition, a comparison elucidating the time saved using the proposed programming method as opposed to simulation-based programming approaches is also discussed. The chapter concludes with a critical evaluation of the technological and economic added value of the new programming methodology.

8.2 Prototype of the APS

The four modules elucidated in the previous chapter were cast into a software prototype with the acronym APoFIS (Automatic Programming of Flexible Inspection System). APoFIS was implemented within the MATLAB® environment using the MATLAB® and native C++programming languages. The benefit of using the MATLAB ® environment was to be able to tap into the vast amount of algorithms available in its large toolbox library. In this manner, implementation time and effort could be reduced. C++ was then used to code time-critical and computationally intensive functionalities that require faster execution, e.g. the virtual sensor, the *scannability map* and the communication interface (see section 8.3). The C++ software components are compiled as dynamic link libraries (dlls) and invoked in the MATLAB® code using the MEX[27] Application 'Programming Interface (API). As MPK—the motion planning kernel—is also implemented in C++, the same MEX API was used to embed the latter in the APoFIS.

As illustrated in figure 8.1, the APoFIS is designed as a three-tier application, composed of a logic layer, a presentation layer and a database layer. The logic layer is built upon the four modules presented in the previous chapters. Each module is implemented as a stand-alone component with encapsulated interfaces. Interconnecting the modules is achieved by stamp coupling, i.e. a composite data structure shared by all components. The presentation layer, in the form of a graphical user interface (GUI), is plugged on the top of the logic tier. It is in charge of graphically

[27] The MEX interface provides the ability to embed C, C++ or FORTRAN code in MATLAB without having to rewrite them as MATLAB files.

presenting the programming process and activities. The GUI is conceived as a window front end that implements bidirectional data exchange from the users to the programming system and back. In this way, trained operators can take control of the programming flow if they want to strike out new parameter settings or reconfigure the FIS environment. The database layer, in comparison, is where data intended to be persistent (e.g. CAD models, DH-parameters, and configuration data) are stored.

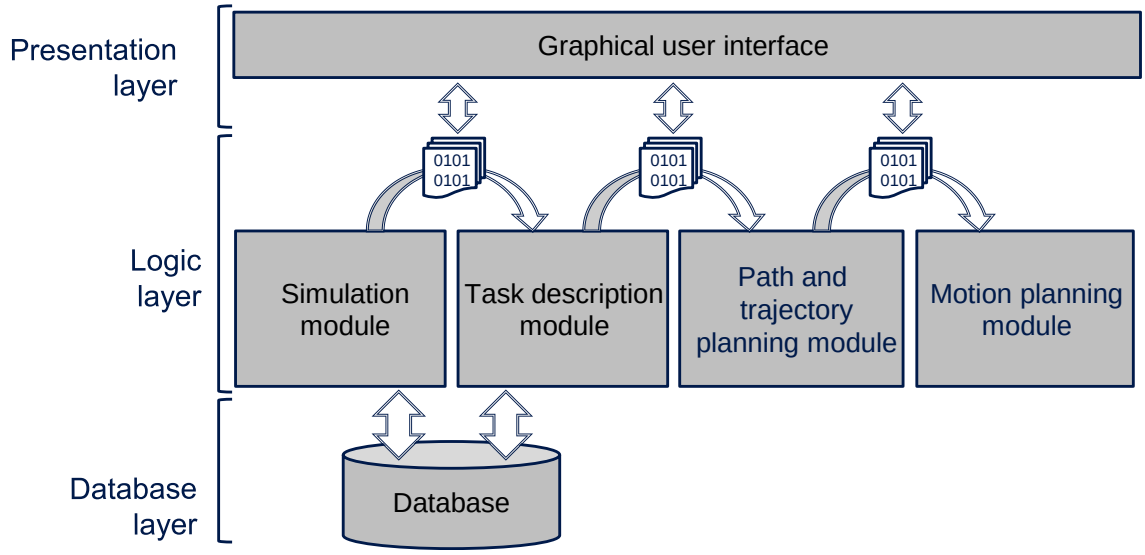


Figure 8.1: Three-tier software architecture of APoFIS

The GUI window includes six tab pages which read or modify the composite data structure mentioned above. The tab pages are labeled "Configuration", "Scannability Map", "Task Definition", "Sensor Trajectory Planning", "Motion planning", and "FIS Interface". The left frame of figure 8.2 depicts the design of the tab page "configuration." This interface allows the user to create, load, display, modify, and save the configuration of the FIS workcell.

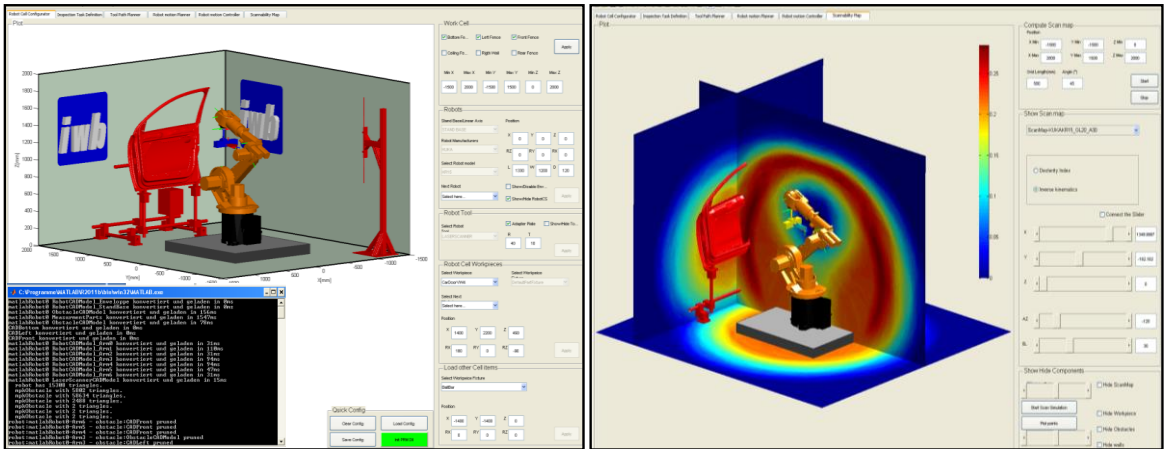


Figure 8.2: Snapshot of the tab pages "Configuration" (left) and "Scannability Map" (right)

In the right frame, the "Scannability Map" tab page that gives the user access to the eponymous functionality is shown. There the user can consult the *scannability map*, and accordingly reshuffle the workcell configuration.

The left-hand side of figure 8.3 pictures the tab page "Task Definition", responsible for displaying the CAD feature contour lines to the user for selection. Choosing the feature lines is carried out either by walking through all features contained in the CAD model one by one or using the mouse to directly mark some feature lines. The tab page on the right-hand side of figure 8.3 represents the user's interface to the sensor path and trajectory module. It displays facilities for editing the constraints related to the sensor path and trajectory. Advanced users can, for instance, tighten the aperture angle of the *scannability cone* to account for optical disturbance caused by ambient light. Also, the locus plots of the translational and orientation scan curve can be displayed upon request.

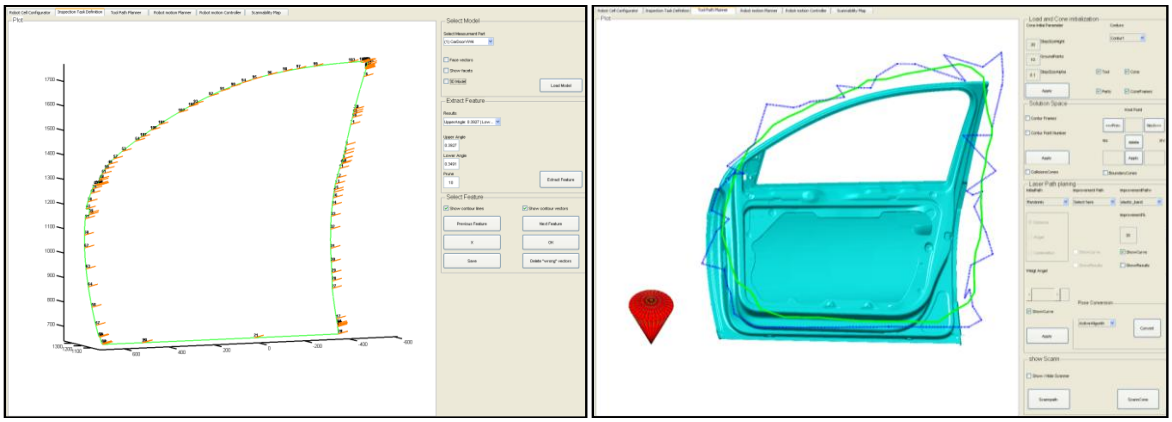


Figure 8.3: a) Snapshot of the tab pages "Task Definition" (left) and "Sensor Trajectory" (right)

The left-hand side of figure 8.4 depicts the tab page "Motion Planning" where the results generated by the motion planning module can be animated and visualized.

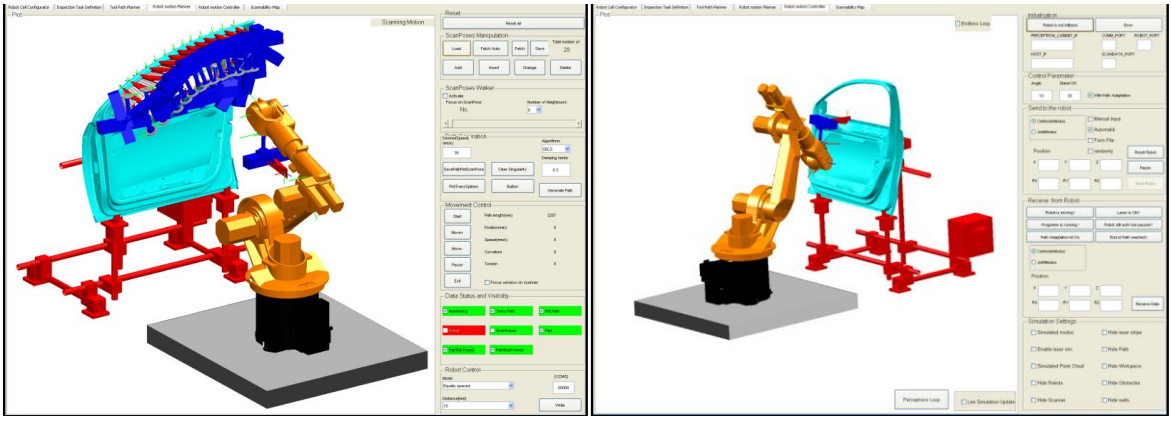


Figure 8.4: a) Snapshot of the tab pages "Motion planning" (left) and "FIS Interface" (right)

The last tab page on the right-hand side of figure 8.4—"FIS Interface"—has the function of transferring scan trajectories to the real FIS. Additionally, data collected by the monitoring interface (to be introduced in section 8.3.2) are displayed.

8.3 Interfacing with the Real FIS

8.3.1 Communication Interface

A schematic representation of the communication architecture required by APoFIS is depicted in figure 8.5. The main components of this architecture are a master PC, the manipulator controller unit, the sensor controller, and the communication link. The master PC is a Windows-based computer capable of running the APoFIS. As Windows operating systems are known to be non-deterministic, all time-critical software pieces run in high-prioritized threads to guarantee the semblance of deterministic behavior. In the remainder of this section, some light will be shed on the software, hardware, and data interface of the communication architecture.

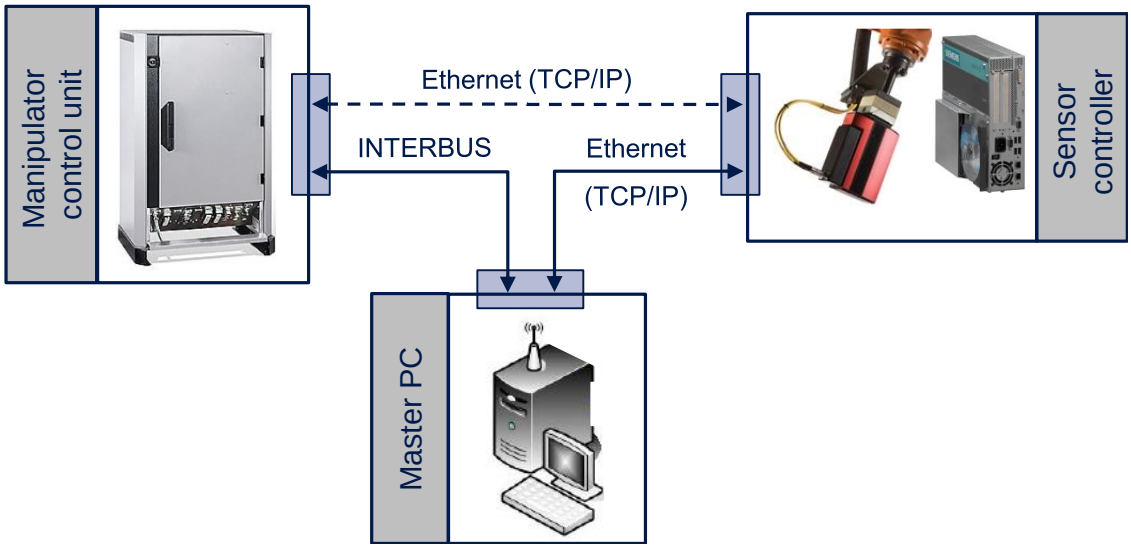


Figure 8.5: Communication architecture required by the APoFIS.

Software Interface

The software application that brings the hardware architecture into action consists of three components: one that runs on the master PC; another which is hosted in the robot controller unit; and a third component that is located on the sensor controller. The software component hosted in the manipulator's controller unit has two software tasks: a high-prioritized task that runs in the foreground and initiates or cancels the manipulator movements, and a low-prioritized task that continuously handles the data traffic to and from the master PC. This data traffic is conveyed

through two channels: a non-real-time channel, dedicated to the transmission of sized data packages (e.g. trajectory data or monitored data) and the real-time channel, which is solely reserved for data traffic related to the visual servoing system. Data arriving through the quasi real-time channel are directly conveyed to the Robot Sensor Interface (RSI). The latter is a path correction mechanism that allows a trajectory currently being executed by the manipulator to be adjusted online by means of a 6D offset feed at execution time (RICHTER & LUETH 2010). Data which arrive through the non-real-time channel are parsed and saved in a FIFO (First In, First Out) data structure, which is continuously processed by the foreground task. Data consistency within the structure is ensured through handshakes. The software component hosted on the sensor controller also implements a quasi real-time data channel to the master PC based on a client-server communication. Through this channel, the sensor readings are tunneled to the master PC upon request. Trigger signals to switch the sensor on and off are also sent from the master PC.

Hardware interfaces

The communication between the master PC and the manipulator controller is based on Ethernet TCP/IP. The data traffic between the master PC and the manipulator controller is enabled with an INTERBUS fiber-optic based communication link. INTERBUS termination boards are plugged into the master PC and the robot controller to enable them to communicate via this link.

8.3.2 Monitoring Interface

The monitoring interface is a piece of software hosted on the master PC that exploits the communication interface to perform automated data visualization and data logging. For this purpose, the interface allows the FIS to be paused and stopped, to record and play back realized trajectories. As it is not improbable that the physical FIS will acts diversely in the reality as compared to the simulation stage, this interface is endowed with the ability to issue an emergency stop in case of pending collision. This task is handled by a watchdog[28] thread that is slaved to the manipulator's encoder. The watchdog continually feeds the GUI, which in turn, updates the virtual FIS to make it reflect the state of its physical counterpart. After each update, the collision engine is queried to determine the closeness of the virtual FIS to the WP and the fixturing device it is held in. For performance reasons, only interference between the FIS, WP and fixturing device, are monitored.

[28] A watchdog is a software thread that is periodically called by an operating system.

8.3.3 Data Transfer Rate

The update rate supported by the real-time channel routed through the INTERBUS is an average of 24 ms. This relatively low sample-time, compared with the intrinsic INTERBUS update rate, is attributable to custom implemented handshakes enforcing proper communication. The bandwidth available on the second quasi real-time channel is faster. Sensor data queries through the Ethernet-based channel can be achieved at 20 ms time intervals. To prevent synchronization troubles, both channels are clocked with the rate of the slower one, i.e. that of the INTERBUS.

8.4 Experimental Platform

To adapt theory to reality, the performance achievement of the APoFIS will be demonstrated in a test rig similar to those FISs' workcells deployed in industrial applications. Figure 8.6 gives an overview of the experimental setup built at the Institute for Machine Tools and Industrial Engineering of the TU Munich (*iwb*) for this purpose.

Figure 8.6: Experimental platform

8.5 Real Case Study: Car Door Inspection

In this section, the application of APoFIS in an industrial case study is presented. The demonstration scenario consists of conducting an inspection task on an automotive sheet metal WP. This demonstration scenario was chosen in partnership with the metrology solutions turnkey supplier Perceptron Inc., which partially initiated and supported this research. They justified their choice by arguing that car doors are the types of WPs that challenge their metrology specialists and keep them

busy onsite. The demonstration consisted of capturing three quality-relevant features of the car door as illustrated in figure 8.7.

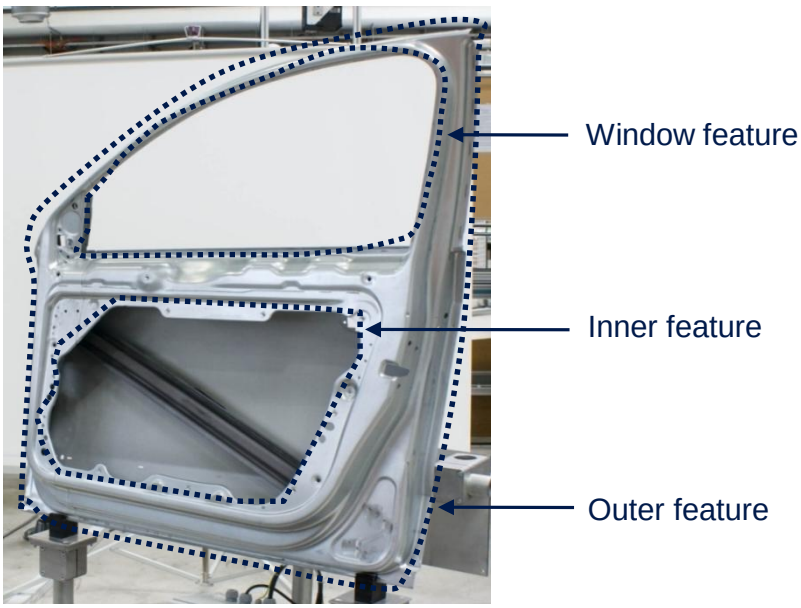
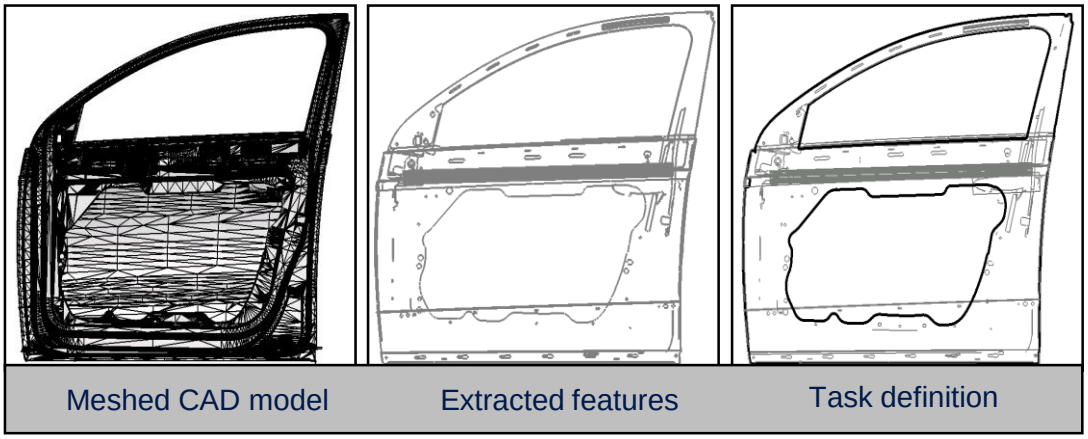
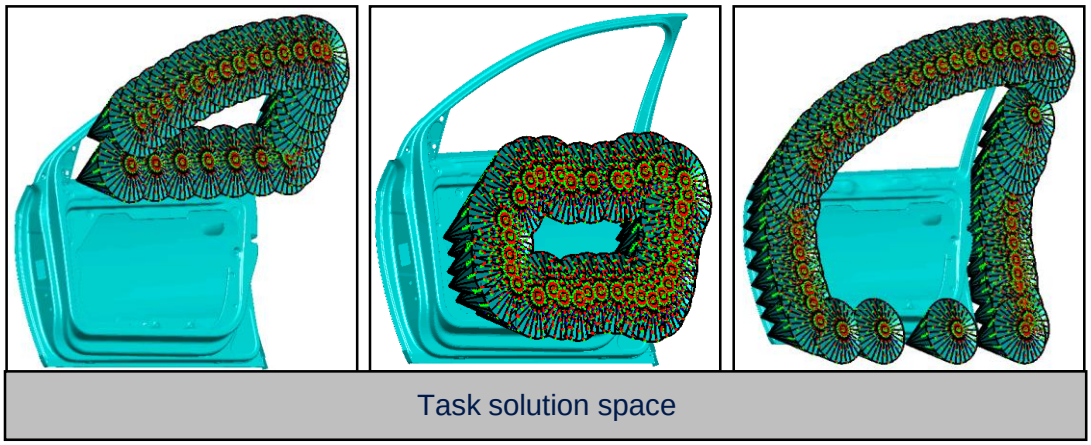
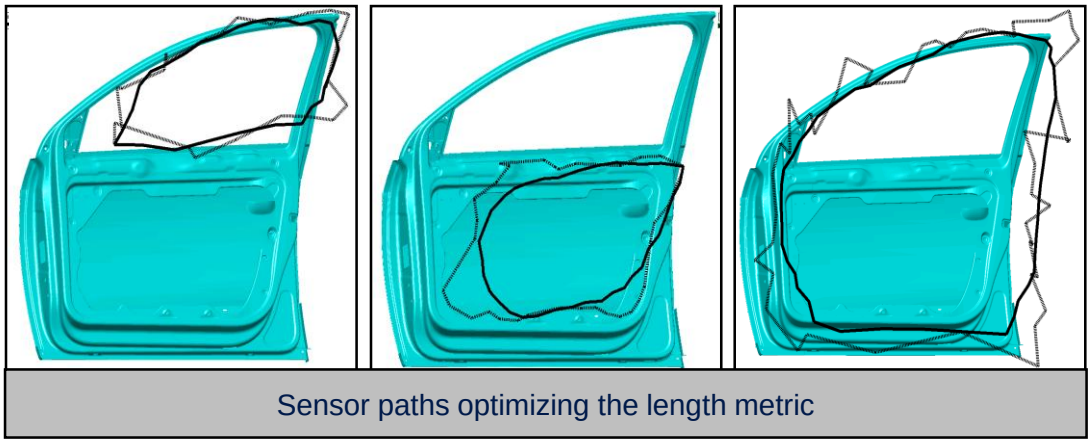


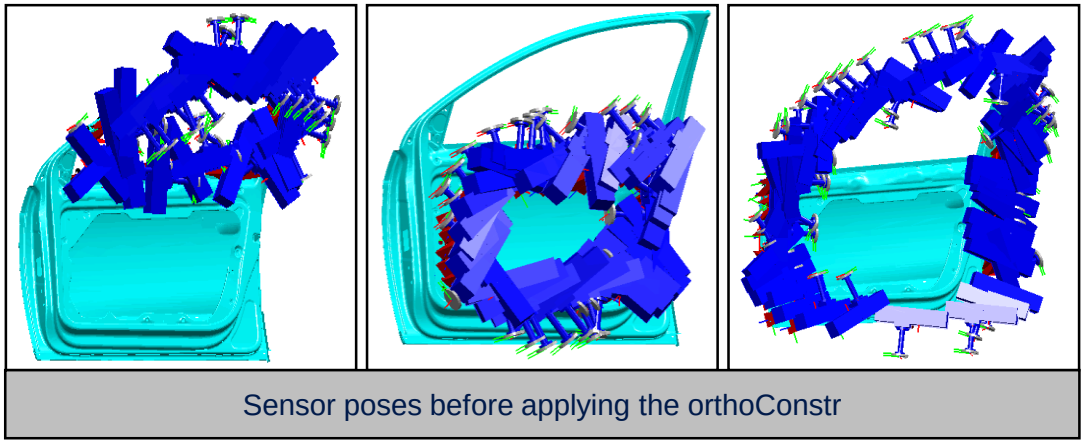
Figure 8.7: Quality-relevant features to be measured on the car door

Each of these contours exhibits a different level of programming complexity. While the "window feature" is relatively simple, the "inner feature" and "exterior feature" are more challenging: the former due to its curved profile, and the latter because of its large size. The step-by-step results issued by the APoFIS during the experiment are summarized in the image sequence of figure 8.8 (see the next pages). As the programming steps have been detailed in the proceedings chapters, additional explanations are omitted for reasons of brevity.

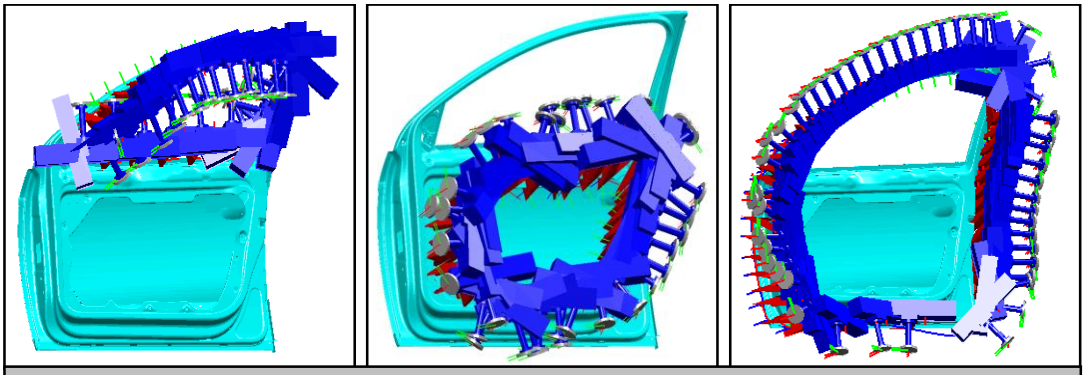

Meshed CAD model
Extracted features
Task definition


Task solution space

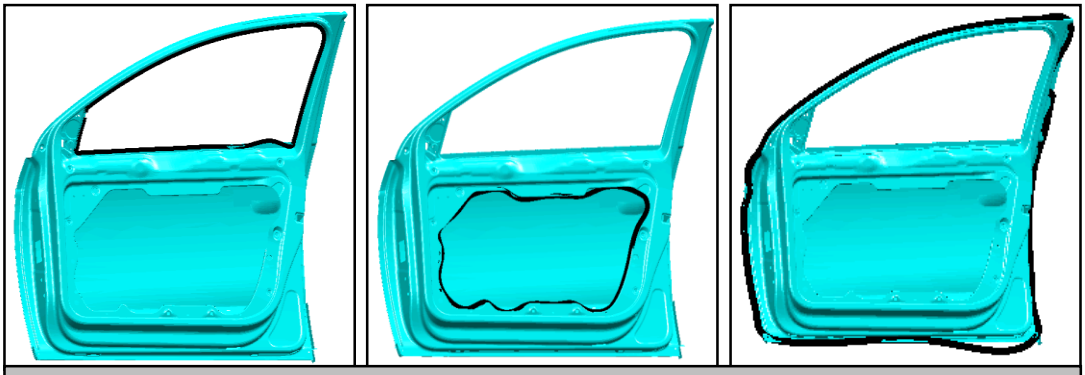

Sensor paths optimizing the length metric


Sensor poses before applying the orthoConstr

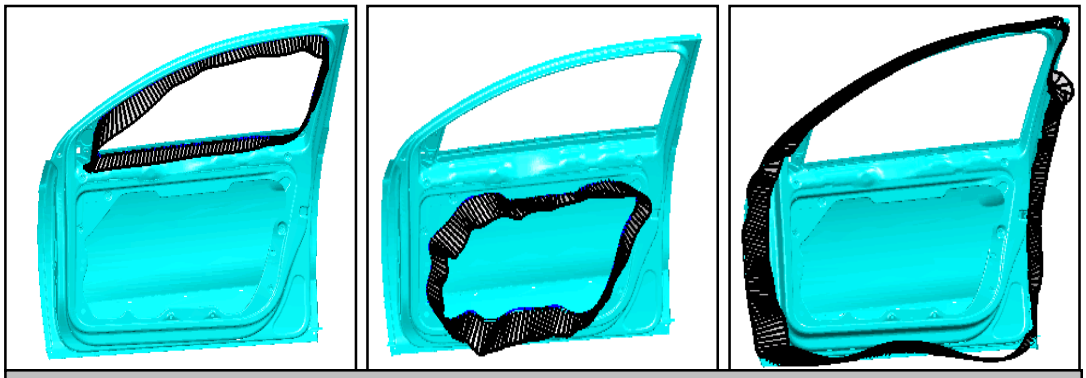
8.5 Real Case Study: Car Door Inspection



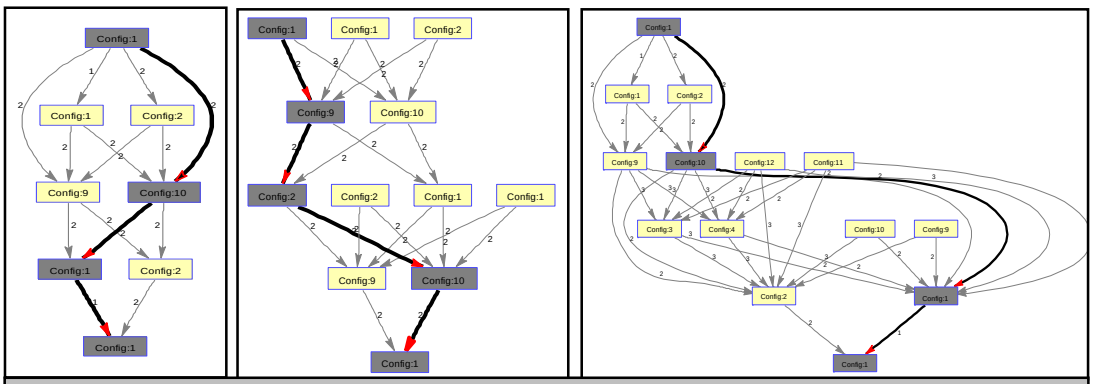
Sensor poses after the orthoConstr is applied



Smoothed translational scan curve



Smoothed orientational scan curve



Optimizing the configuration changes

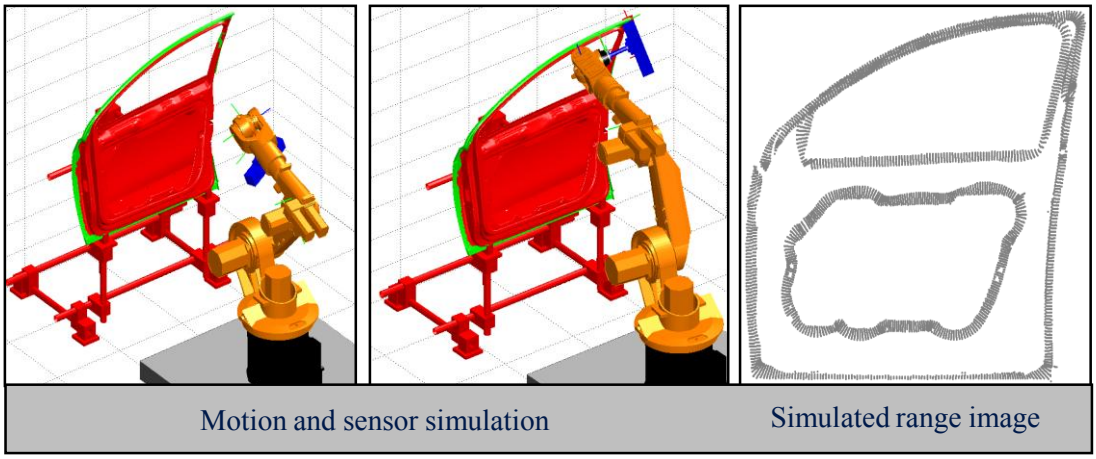


Figure 8.8: Step-by-step results issue by APoFIS during the programming of the FIS to perform the real case study

Figure 8.9 documents the various steps of the execution phase. The three first frames show the FIS performing the inspection task. The fourth frame illustrates the digitization results, while the last frame shows the inspection result in the form of a color-shaded image report. The color distribution underscores the deviation between the actual door's shape as manufactured and the nominal dimensions given in the CAD model. The amount of deviation can be read from the color bar.

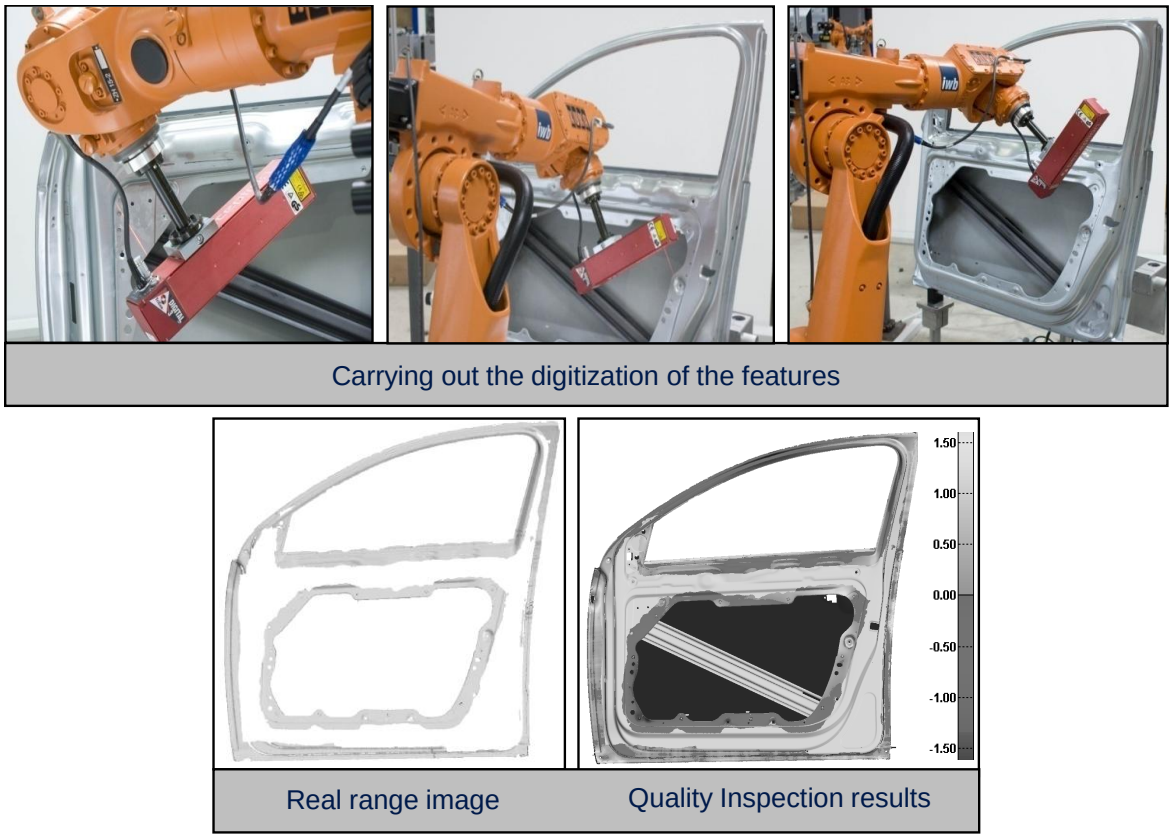


Figure 8.9: Digitization of the selected WP features and QI results

8.6 Comparative Study

To put the advantages of APoFIS into a realistic perspective, a non-representative comparative experiment was conducted, aimed at highlighting its saving potential. The experiment focuses on the comparison of the programming time when using the APoFIS as compared with using a simulation-based programming system. The resulting process cycle time is also put in contrasted. It is important to note at this juncture that simulation-based programming is the approach by which FISs are usually programmed. The comparative experiment was conducted by an experienced metrologist from Perceptron Inc. and consisted of programming the experimental setup presented in section 8.4 to perform the inspection task formulated in the real case study. The decision to involve an expert as tester was motivated by the fact that deep technical knowledge is required to carry out a simulation-based programming task on a FIS.

During the comparison study, the programming task was divided into four subtasks and the time spent on each of them recorded. The range image-to-CAD comparison is intentionally ignored because the time spent on this is expected to be the same for both programming approaches. The tester was given a 20-minute introduction to the functionalities of the APoFIS before commencing the experiment. Table 8.1 contrasts the results obtained during the experiment. The figures related to the simulation-based programming were not obtained directly on the experimental platform but rather on a similar FIS deployed at the Opel car plant in Ruesselsheim. In view of the tester's restricted time availability, it was not possible to perform the simulation-based programming directly on the experimental platform.

Table 8.1: Figures obtained during the comparative study

Programming step	**Simulation-based**	**APoFIS**
Workcell modeling [min]	30	20
Task definition [min]	120	15
Motion programming [min]	75	10
Reteach/troubleshooting [min]	40	20
Total time [min]	265	65
Average [%]	100	25
Cycle time [s]	205	158
Cycle time [%]	100	77

From the results in Table 8.1 it can be inferred that the proposed methodology allows a reduction of the programming time of up to 75% as compared with using a simulation-based system. As for the cycle time, saving an average of 23% can be achieved. According to these results, it can be stated that the general applicability of the programming system for industrial application is proven. By the same token, the improvements accomplishment of the presented methodology in term of easing the effort the applied for programming FIS can be considered as evidenced.

8.7 Technical and Economic Assessment

This section presents a critical evaluation of the technical and economic added value of APoFIS as well as the drawbacks of the general methodology. The discussion will basically be based on the insights gained through the real case study presented in the previous section. The intrinsic advantages resulting from the deployment of the FIS in manufacturing lines will be addressed from a purely qualitative point of view.

8.7.1 Technical Assessment

In the following discussion, the benefits of APoFIS in comparison to simulation-based programming systems, which represent the mostly-utilized programming tool in the context of FISs, are outlined. The comparison is carried out on the basis of six metrics that are often quoted in the literature (ISO/IEC 9126-1 2001, POTINECKE 2010) in connection with evaluating the performance characteristics of CAx[29] software tools, namely *productivity, quality, flexibility, reliability, user-friendliness / user acceptance and synergy.*

8.7.1.1 Productivity

The core benefit of the developed APoFIS is a significant reduction of the programming time and effort, and by implication, the increase of productivity. As demonstrated in the real case study, time savings of up to 75% can be achieved compared with conventional off-line programming systems. Another argument in favor of the newly developed system is the shortening of process cycle times by up to 23% through the diverse optimization algorithms embedded in the system. It is also anticipated that additional time can be saved during the modeling phase of the FIS workcell. This claim is based on the fact that the workcell does not need to be

[29] CAx stands for "Computer-Aided technologies" and is an umbrella term for computer technologies that are utilized in the conception, design, analysis and manufacture of a product.

modeled accurately, since the APoFIS possesses the ability to compensate for modeling inaccuracy.

8.7.1.2 Quality

This research did not attempt to directly ameliorate the outcome of the inspection process in the sense of improving the accuracy of the measurements results. Thus, from a conservative point of view, scan trajectories generated by APoFIS do not contribute to higher measurement accuracy. Nevertheless, the enforcement of the *HiqualConstr* along the scan trajectories leads to qualitatively better range images, which are the premise for achieving expressive QI analysis. Additionally, the reduction of the complexity of programming FIS by APoFIS opens the door to extending the scope of inspection activities manufacturing processes. For instance, intensive measurements in the early phase of product development constitute a knowledge base on which manufacturing process specialists can build to foster their understanding of the interaction between the process parameters. Providing the specialists with extensive measurement data will help them to rapidly eliminate process instabilities that arise in the production ramp-up phase (RASHIDY 2009 p. 131).

8.7.1.3 Flexibility

The high cost of purchasing and servicing production machinery dictates that they be kept working on a continuous basis (BARD 1986) and/or be reused in future manufacturing lines (VERL ET AL. 2010). In the context of small batch production, this presupposes that the equipment can be adapted to accommodate current and future product types in a short time and with reasonable effort. By reducing the effort of programming FISs, the APoFIS precisely makes a valuable contribution toward achieving this goal.

8.7.1.4 Synergy

Synergy can be understood to constitute the indirect benefits of the presented programming approach that can be unfolded in different stages of the product development process. Such additional advantages are expected to arise mainly in the context of Computer-Integrated Manufacturing (CIM). Through its standardized CAD interfaces, the APoFIS can be easily integrated into existing CIM infrastructures to help secure planning-relevant key results, such as downtime period and cycle time. Also, utility tools such as the *scannability map* will help planners determine the suitability of the existing FIS workcell to new inspection tasks. Additionally, it can be argued that the APoFIS sets the stage for a seamless quality

assurance process chain without interface losses. Indeed, the methodology enables the use of the WPs' CAD models as a unique data carrier through all phases of the product development process. In this way, consistent documentation and reproducible information flow is guaranteed.

8.7.1.5 Acceptance

As it is a common wisdom that humans in general are suspicious of change (BARD 1986), one may logically presume that an advanced programming system such as APoFIS will not be welcomed by everyone. This suspiciousness may be the result of fear and uncertainty on the side of the stakeholders. Therefore, by pointing out the benefits and advantages to them, their opposition is likely to disappear, or at least be tempered. APoFIS stakeholders include first and foremost the management of the manufacturing company, and the system operators. The interest of the first group in introducing the APoFIS is obvious: reduction of human labor, shortening of training time, growing productivity, and short payback period. As the APoFIS exhibits strong arguments in favor of these interests there is little resistance to be expected from this side.

Trained operators, e.g. programming experts, in contrast, must be dealt with tactfully. They are likely to be defiant about the operating efficiency of APoFIS or worried about long-term deskilling or the possible loss of their jobs. To combat such negative stereotypes, they should be granted the upper hand over APoFIS at all times. They could also be involved in the acquisition decision process. As for untrained operators, they may perceive the APoFIS as a complex, user-unfriendly system they have been assigned to work with. Therefore, issues such as who is responsible for the technical maintenance of the APoFIS should be clarified as early as possible to prevent a bad feeling among the operator crew from the start. In addition, illustrated manuals and examples, training seminars and a user-friendly interface are all features that could encourage operators to rapidly embrace such a new programming system.

In more general terms, management should be aware of the attitude of their key technical employee toward "intelligent software" and/or "autonomous systems" well before the intention of deploying any such system goes public. If necessary, mitigation strategies should be adopted in advance to circumvent any resistance.

8.7.1.6 Reliability

The programming methodology embedded in APoFIS was vetted in several experiments carried out on the experimental platform presented in section 8.4. From

the observations made during these experiments, the following key liability drivers could be traced:

Modeling uncertainties: Because of modeling uncertainties, there is always the danger that the physical FIS behaves differently to its virtual counterpart in the programming system. As a consequence, the generated trajectories may lead the FIS to damaging interference with some workcell components. On that account, shorter maintenance loops with regards to the environment model may be necessary to keep the system running.

System complexity: The integration of additional hardware components into the FIS architecture may adversely affect the system reliability. Moreover, because the visual servoing loop engages at a fairly low-level interface in the manipulator controller unit, the overall complexity of the overall system is driven up. It is therefore legitimate to expect the system's statistical probability of failure to be downgraded. Hosting the visual servoing system directly into the robot control unit may drop the complexity issue to a reasonable level.

8.7.1.7 Conclusion

To summarize the technical analysis of APoFIS, its advantages as compared with simulation-based programming systems are graphically highlighted in figure 8.10. This figure shows that the APoFIS outperforms simulation-based programming systems with regards to all performance characteristics except reliability.

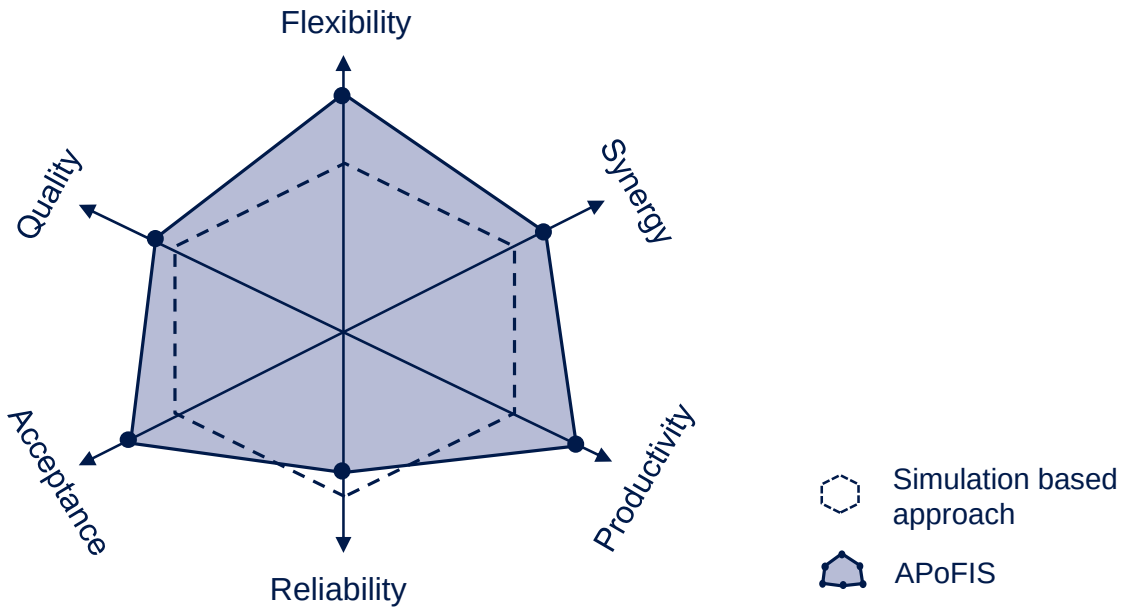


Figure 8.10: Qualitative comparison between the proposed system and simulation-based systems

8.7.2 Economical Assessment

It has by now become a matter of common knowledge that the establishment of a new technology does not only depend on its technical finesse. Just as pertinent is the return on the financial investment necessary for the deployment of the new technology, as well as the associated running costs. Consequently, this section addresses the question of when the benefits of using the APoFIS are expected to exceed the total cost of ownership. The calculation method used is, cost-benefit analysis, since it lends itself perfectly to such an endeavor (KRUSCHWITZ 2009 pp. 33–35). The premises that supplied the basis for the calculation are the following:

- The investment cost needed to deploy the programming system is estimated at € 25,000, with software costs accounting for the biggest share of the expenditure (€ 20,000). Hardware costs arise relating to the purchase of a high-end master PC on which the programming system run. Ordinary office-PCs are inadequate because they furnish the required computational power.
- It is assumed that the FIS system is used 48 weeks per calendar year. During this time span, the FIS is reprogrammed twice a week to accommodate product changeovers. A programming time of
- Using APoFIS, an operator will, on average (re-)program the FIS 75% faster when compared with using a simulation-based programming system. Thus, one programming hour using a conventional programming system corresponds to a $(1 - 0.75) \times 60 = 15$ min utilization of APoFIS only.
- Longevity and freedom from early obsolescence of the FIS are assumed.

Table 8.2 gives a detailed breakdown of the costs and benefits related to the utilization of APoFIS under the premises listed above. As can be seen, the annual beneficial effects resulting from the use of APoFIS are valued at a total € 23,040. This positive impact is offset by additional annual expenditures amounting to € 9,125. In sum, the break-event point is estimated to be hit after about 1.3 years.

It is imperative to note that this economic assessment does not take account of the benefits of intangible assets. For instance, the opportunity for higher quality makes resulting from a widespread use of FISs. Furthermore reduced operator training time and improved customer satisfaction are neither explicitly nor implicitly accounted for in the analysis. Also, the impact resulting from the process cycle reduction has not been considered. Therefore, it is obvious that the payback time calculation carried out above is based on stringent premises. In most cases, the break-even point will be meat earlier than the estimated 1.3 years. On that account, acquiring APoFIS can be regarded as a valuable investment, even if the calculation premises are shifted toward less favorable figures.

Before closing this chapter, it is worth mentioning that any investment always involves a technical and economic risk that cannot be eliminated. The appraisal given in this section solely seeks to reduce the accompanying uncertainty and to serve as a monetary indicator of the magnitude of cost that is to be expected when some basic conditions hold true.

Table 8.2: Cost benefit analysis of APoFIS

Initial investment required for APoFIS		
Software costs		20,000 €
Hardware costs		3,000 €
Cabling, engineering costs, installation		2,000 €
Investment Costs	I	25,000 €
Annual cost occasioned by APoFIS		
Investment costs	I	25,000 €
Utilization period	P	5 a
Annual depreciation cost (linear)	D = I / P	5,000 €/a
Interest rate	R	9 %
Calculator interest rate	K = Z x I/2	1,125 €/a
Additional expenditure (maintenance, energy, training, indirect cost)	A	3,000 €/a
Total annual cost	F = D + K + A	9,125 €/a
Annual benefits resulting from the use of APoFIS		
Hourly cost of programming labor	P	80 €/h
Procentual time saving	t	75 %
Labor saving	L = t/(1-t)	3 h/h
Utilization time	N	96 h/a
Saving effect	S = P * L * N	23,040 €/a
Amortization		
Payback period	T = I / (S - F + D)	1.32 a

9 Summary and Future Research

9.1 Summary

In the sheet metal industry, as the number of product variants explodes, maintaining high-quality standards is becoming more and more crucial. Sheet metal part manufacturers cannot afford the commercial absurdity of shipping poor quality products to the consumers. After all, "quality is remembered long after the price is forgotten."[30] To avoid the unpredictable consequences of their trade brand being linked to quality troubles, manufacturers strive to achieve zero-fault production. A major step toward this goal is the early identification and rectification of quality issues.

As argued in the introduction of this research, flexible inspection systems (FISs) represent powerful metrology devices that can be used for this purpose. A flexible inspection system consists of a robot manipulator equipped with a short range, high-accuracy light-section sensor. This sensor uses the principle of optical triangulation to measure variations in distance between the sensor probe and the workpiece and, in this way, to digitize its surface. Quality blemishes in terms of dimensional non-conformity are then determined by comparing the digitization data with the nominal CAD model of the workpiece under investigation.

Nevertheless, despite their technological advantages, the potential of FISs is still far from being exhausted. This is mainly due to the significant programming effort necessary to adjust them to changing inspection subjects. Indeed, (re)programming FISs involves time-consuming activities due to the multitude of constraints that must be simultaneously observed for the system to digitize the features of interest on a workpiece. Those programming activities take up a large portion of the change over activities and unnecessarily prolong equipment downtime. In addition, programming results heavily depend on the operator's experience, e.g. his exposure to robotics and knowledge of metrology. Taken together, these drawbacks currently negatively impact the cost-effectiveness of FIS and hamper their dissemination in manufacturing lines, especially in those dedicated to small batch production.

A review of existing works was unable to identify existing solutions that properly addressed the issue of easing the programming effort related to FISs. Accordingly, this doctoral dissertation focuses on the task of reducing the time and the specialized knowledge required to (re)program FISs. To accomplish this endeavor, a task-level programming methodology that shields the operator from complex details was devised. The basic idea behind this approach is to specify the digitization at a high

[30] This saying is attributed in some literatures to STANLEY MARCUS whereas others accredit it to the GUCCI FAMILY which used it as a selling slogan.

level of abstraction using a workpiece CAD model and leave it to dedicated algorithms to fill in the details of how the FIS should best perform the task. The core steps of this new methodology can be summarized as follows:

- CAD-based extraction of the workpiece design features;
- Feature-based description of the digitizing task;
- Automated generation of the scan paths and trajectories which allow the sensor to capture the selected features;
- Automatic synthetization of suitable movements of the FIS that steer the sensor along the generated scan trajectories;
- The online adaptation of the scan trajectories at execution time to accommodate with the mismatch between the real and virtual worlds.

An automatic programming system that implements the methodology steps listed above is developed and explained in detail. For this purpose, the methodology is broken down into four modules: *a simulation, an inspection task description, a path and trajectory planning* and *a motion planning module*.

The *simulation module* involves a virtual model of the FIS workcell and serves as a knowledge base for the programming algorithms. Another essential feature of this module is the virtual sensor, an emulation of the real light-section sensor that can reveal the quality of the results generated by the programming system even before the physical FIS is switched on. The *task description module* encapsulates algorithms that extract feature contour lines from the CAD model of a given workpiece. The algorithms implemented in this module were adapted from the realm of computer vision. To describe the inspection task, the operator limits himself to choosing one of more of the extracted features. As the position of the workpiece with respect to the FIS determines the ability of the FIS to perform the digitization task, a layout-planning tool labeled *scannability map* is provided in the same module. This tool visually indicates to the operator where the FIS can best access a measurement object.

The *sensor plan and trajectory* module operates by first determining an ordered sequence of manipulator-independent sensor viewpoints that ensure an efficient and robust coverage of the targeted features. Robust coverage means that specific constraints which allow the sensor to adequately capture the targeted features must be met by each of the viewpoints. Finally, to fully describe the state of the sensor, the viewpoints are interpolated using cubic splines and augmented with a time law to form scan trajectories. The *motion planning module* is aimed at ensuring that the manipulator can steer the sensor along the computed scan trajectories. To this end, this module is endowed with algorithms that suppress collisions or singularity handicaps that the FIS may encounter during the realization of the scan trajectories.

In addition, further algorithms geared toward the optimization of the process cycle time are also available. The module also features a visual servoing system that relies on the sensor reading at execution time to control the standoff distance between the sensor head and the workpiece surface. In this way, the mismatches between the real and virtual worlds resulting from inaccurate models are compensated for in such a way that the sensor field-of-view always remains in focus.

To prove the efficacy of the programming system, these modules were cast in a software prototype (APoFIS) and tested on an industrial platform. An inspection job specified by a turnkey metrology systems provider with firsthand experience of the complexity of programming FISs was used as a reference scenario. The experimental results revealed that using the software prototype, the task of programming FISs could be carried out four times faster than using conventional simulation-based systems. Also, the process cycle time could be reduced by up to 23%. These results confirm the high potential to accelerate and simplify the programming of FISs using the proposed approach.

The technical assessment discussed in the previous chapter unveiled the advantages of the proposed solution as compared to simulation-based programming systems by means of which FISs are currently programmed. Additionally, the insights elaborated there endorsed a further objective of this thesis, namely that of bringing the task of programming FISs within reach of operators without technical experience in metrology or robotics.

Looking at the bigger picture, one may be quick to observe that the proposed solution may save valuable time during the product development process and set a milestone toward the implementation of a seamless quality assurance chain without interface losses. It may also be regarded as a door opener for the deployment of FISs by companies that until now were reluctant to acquire such systems because of the high cost of ownership. The relatively short payback time period of one year is a figure that effectively mirrors the cost-attractiveness and profitability of the proposed solution.

To conclude this summary, it should be noted that while the methodology presented in this work may have been tailored for FISs, it is nonetheless transferable with a corresponding amount of effort to contour-based robotic applications such as arc-welding, bonding and sealing applications.

9.2 Future Research and Open Issues

For future development of the proposed solution, five major aspects can be identified:

The *first aspect* regards the enhancement of the virtual sensor with a light-reflection model that describes the interaction of the laser light with the workpiece surface. In a second step, an illumination model could also be implemented to properly describe the effects of indirect light (e.g. ambient light) on the workpiece surface. In doing so, the virtual sensor will better mimic its real counterpart and, that way, open the doors for planning more accurate scan trajectories. For instance, if the workpiece has a shiny surface, such a model could be helpful at the viewpoint selection stage to discard sensor poses subjected to direct reflection. Such an accurate sensor model may also prove useful for capturing sharp edges that often cause the laser beam to "ricochet" and thus to create false geometry (GELFAND 2006 p. 101).

The *second aspect* that may lead to considerable improvements of the proposed solution could be the use of the sensor focal length as an additional DoF at the planning phase. In the proposed solution, the camera focus point is considered a fixed parameter to be set once during the commissioning of the FIS. Actually, light-section sensors are equipped with interfaces that allow tuning the camera focal length during operation. Relaxing the restriction on the focal length frees up an additional variable parameter that can be exploited by planning algorithms to drive the FIS around collisions, singularities and undesired configuration changes. This variable parameter may also offer a wider flexibility during the selection of the sensor viewpoints.

An aspect that may represent an interesting development of the proposed approach could also be the extension of the programming methodology to accommodate multiple FISs workcell. As the time to complete a digitization task may be comparatively high depending on the amount of features to be collected, it may be advantageous to consider multiple FISs for carrying out extensive inspection tasks. In some cases, the incentive for doing so may go well beyond productivity. For example, important savings in production space can be brought about by merging different FISs into one single cell as compared to the complete footprint of distributed single FIS workcells. However, more than one manipulator working in a shared workspace operating simultaneously but asynchronously in close proximity raises additional issues. Chief among these are collision avoidance, efficient workload balancing and synchronization. Besides, to underscore the complexity of this undertaking, it should be noted at this juncture that even if the trajectories of the respective FISs are specified, synchronizing them in a common workspace is an NP-hard problem (SRINIVAS & SETH 2002, CARPIN & PAGELLO 2009).

Another aspect worth being investigated is the extension of the programming methodology with a workpiece localization system. This means equipping FISs with long range vision sensors that are assigned the task of locating the workpiece within the workcell. These additional sensors may be rigidly fixed or fastened to the

robot for greater flexibility. Improving the proposed solution that way is expected to further reduce the modeling and programming effort of FISs.

Finally, an *open issue* that should also be the subject of future researches is the broadening of the class of workpiece that can be processed by the proposed approach to 3D solid parts with more general geometries.

10 Bibliography

ABB 2005

The ABB Group: ABB RobotWare IRC5. < http://www.abb.de/product/seitp327/3e92592d974a333cc125737c00373449.aspx?productLanguage=ge&country=DE&tabKey=7> - 23.06.2011.

ABBATE 2005

Abbate, S.: Introducing the Next Generation of Hemming and Measurement Tools.<http://www.hirotecamerica.com/images/2005_next_generation.pdf> - 03.08.2010.

ABDERRAHIM ET AL. 2007

Abderrahim, M.; Khamis, A.; Garrido, S.; Moreno, L.: Accuracy and Calibration Issues of Industrial Manipulators. In: Low, K.-H. (Eds.): Industrial robotics. Mammendorf: plV pro-Literatur-Verlag 2007, pp. 131–146. ISBN: 978-3866112865.

ALMEIDA ET AL. 2006

Handhabungstechnik und Robotik (ab 1980). In: Eversheim, Walter; Pfeifer, T.; Weck, M. (Eds.): 100 Jahre Produktionstechnik: Berlin, Heidelberg, New York: Springer 2006, pp. 515–528. ISBN: 978-3540333166.

ANGELES 2007

Angeles, J.: Fundamentals of robotic mechanical systems: theory, methods, and algorithms. 3rd ed., New York: Springer 2007. ISBN: 978-0387294124.

ANTONELLI ET AL. 2003

Antonelli, G.; Chiaverini, S.; Fusco, G.: A new online algorithm for inverse kinematics of robot manipulators ensuring path tracking capability under joint limits. IEEE Transactions on Robotics and Automation 19 (2003) 1, pp. 162–167.

APPLEGATE ET AL. 1998

Applegate, D.; Bixby, R.; Chvátal, V.; Cook, W.: On the Solution of Traveling Salesman Problems: In: Proceedings of the international Congress of Mathematicians. Berlin, Germany, August 18-27, 1998, pp. 645–656.

APPLEGATE ET AL. 2006

Applegate, D.; Bixby, R.; Chvátal, V.: The traveling salesman problem. Princeton: Princeton University Press 2006. ISBN: 978-0691129938.

ATTALI ET AL. 2009

Attali, D.; Boissonnat, J.-D.; Edelsbrunner, H.: Stability and Computation of Medial Axes - a State-of-the-Art Report. In: Farin, G.; Hege, H.-C; Hoffman, D.; Johnson, C.; Polthier, K. (Eds.): Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration: Springer Heidelberg 2009, pp. 109–125. ISBN: 978-3540499268.

BACHILLER ET AL. 2007

Bachiller, M.; Cerrada, C.; Cerrada, J.: Designing and Building Controllers for 3D Visual Servoing Applications under a Modular Scheme. In: Low, K.-H. (Eds.): Industrial robotics. Mammendorf: plV pro-Literatur-Verlag 2007, pp. 81–106. ISBN: 978-3866112865.

BARD 1986

Bard, J.: An assessment of industrial robots: Capabilities, economics, and impacts. Journal of Operations Management 6 (1986) 2, pp. 99–124.

BARRAL ET AL.

Barral, D.; Perrin, J.-P; Dombre, E.; Liégeois, A.: Simulated Annealing Combined with a Constructive Algorithm for Optimising Assembly Workcell Layout. The International Journal of Advanced Manufacturing Technology 17 (2001) (8), pp. 593–602.

BAUM 2004

Baum, E.: What is thought? Cambridge, Massachusetts: MIT Press 2004, pp. 21. ISBN: 978-0262025485.

BRANDENBURG & DILTHEY 2006

Brandenburg, A.; Dilthey, U.: Schweißtechnische Fertigungsverfahren. 3rd ed., Düsseldorf: VDI-Verlag 2006. ISBN: 978-3540216735.

BELLMAN 1962

Bellman, R.: Dynamic Programming Treatment of the Travelling Salesman Problem. Journal of the Association for Computing Machinery (JACM) 9 (1962) 1, pp. 61–63.

BENNETT 1993

Bennett, S.: A history of control engineering, 1930-1955. Stevenage, Herts, U.K: Peter Peregrinus, on behalf of the Institution of Electrical Engineers, 1993, p. 48. ISBN: 978-0863412998.

BERBYUK ET AL. 2010

Berbyuk, V.; Kamrani, B.; Wäppling, D.; Feng, F.; Andersson, H.: Optimal Usage of Robot Manipulators. In: Jimenez, A. & Hadithi, B. (Eds.): Robot Manipulators Trends and Development. InTech (Open Access publisher of books and journals), 2010, pp. 1–26. ISBN: 978-9533070735

BERNARD & VÉRON 1999

Bernard, A.; Véron, M.: Analysis and Validation of 3D Laser Sensor Scanning Process. CIRP Annals - Manufacturing Technology 48 (1999) 1, pp. 111–114.

BERNHARDT & ALBRIGHT 1993

Bernhardt, R.; Albright, S. L.: Robot calibration. London: Chapman & Hall 1993. ISBN: 978-0412491405.

BEYER & WULFSBERG 2004

Beyer, L.; Wulfsberg, J.: Practical robot calibration with ROSY. Robotica 22 (2004), pp. 505–512.

BHANDARKAR & NAGI 2000

Bhandarkar, M.; Nagi, R.: STEP-based feature extraction from STEP geometry for agile manufacturing. Computers in Industry 41 (2000) 1, pp. 3–24.

BI & LANG 2007

Bi, Z.; Lang, S.: A Framework for CAD- and Sensor-Based Robotic Coating Automation. IEEE Transactions on Industrial Informatics 3 (2007) 1, pp. 84–91.

BIAGIOTTI & MELCHIORRI 2008

Biagiotti, L.; Melchiorri, C.: Trajectory planning for automatic machines and robots. Berlin, Heidelberg: Springer 2008. ISBN: 978-3540856283.

BIGGS 2007

Biggs, G.: Designing an application-specific programming language for mobile robots. Doctoral dissertation, University of Auckland, Australia. <https://researchspace.auckland.ac.nz/bitstream/handle/2292/702/02whole.pdf?sequence=5> - 21.08.2011.

BLAIS 2004

Blais, F.: Review of 20 years of range sensor development. Journal of Electronic Imaging 13 (2004) 1, pp. 231–243.

BLEY ET AL.

Bley, H.; Bernardi, M.; Franke, C.; Seel, U.: Process-based assembly planning using a simulation system with cell calibration. In: Proceedings of the IEEE International Symposium on Assembly and Task Planning. Fukuoka, Japan, Mai 28–29, 2001, pp. 116–121.

BLUM 1967

Blum, H.: A Transformation for Extracting New Descriptors of Shape. Models for the Perception of Speech and Visual Form (1967), pp. 362–380.

BOILLOT & UOTA 2002

Boillot, J.-P.; Uota, K.: Flexible robotic measuring of weldments on production lines. Industrial Robot: An International Journal 29 (2002) 1, pp. 43–48.

BONGARDT 2004

Bongardt, T.: Methode zur Kompensation betriebsabhängiger Einflüsse auf die Absolutgenauigkeit von Industrierobotern: Doctoral dissertation, TU München (2004). München: Herbert Utz Verlag 2004. ISBN: 978-3831603-329.

BOSCH 1995

Bosch, J.: Coordinate measuring machines and systems: Manufacturing engineering and materials processing 42. New York: Dekker 1995. ISBN: 978-0824795818.

BRECHER ET AL. 2006

Brecher, C.; Schroeter, B.; Matthias, B.; Kuerzel, A.; Herchel, M.: Portable Robot Systems for Machine Tending Tasks. In: (Eds.): Proceedings of the 37th International Symposium on Robotics. Mai 15.–17., 2006, München, Germany 2006.

BRECHER ET AL. 2010

Brecher, C.; Roßmann, J.; Schlette, C.; Herfs, W.; Ruf, H.; Göbel, M.: Intuitive Roboterprogrammierung in der automatisierten Montage. wt Werkstattstechnik online 100 (2010) 9, pp. 681-686.

BRONSTEIN & SEMENDJAJEW 2001

Bronstein, I.; Semendjajew, K.: Taschenbuch der Mathematik. 5th ed., Frankfurt am Main: Wissenschaftlicher Verlag Harri Deutsch GmbH 2001. ISBN: 978-3817120154.

CALLIERI ET AL. 2004

Callieri, M.; Fasano A.; Impoco G.; Cignoni P.; Scopigo R.: RoboScan: an Automatic System for Accurate and Unattended 3D Scanning: Proceedings of the 2nd IEEE International Symposium on 3D Data Processing, Visualization and Transmission. Thessaloniki, Greece, September 6–9, 2004, pp. 805–812.

CAMPANELLA 1999

Campanella, J.: Principles of quality costs. 3rd ed. Milwaukee Wisconsin: ASQ Quality Press 1999. ISBN: 978-0873894432.

CANNY 1986

Canny, J.: A Computational Approach to Edge Detection. IEEE Transac tions on Pattern Analysis and Machine Intelligence PAMI-8 (1986) 6, pp. 679–698.

CARVALHO 1997

Carvalho, G.: An adaptive Control System For Off-line Programming in Robotic Gas Metal Arc Welding. Doctoral dissertation, Cranfield University, United Kingdom <https://dspace.lib.cranfield.ac.uk/bitstream/1826/4597/1/G_C_Carvalho_Thesis_1997.pdf> - 23.06.2011.

CEDERBERG 2004

Cederberg, P.: On Sensor-Controlled Robotized One-off Manufacturing. Doctoral dissertation, Lund University, Sweden (2004). <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.5347&rep=rep1&type=pdf> - 05.11.2010.

CHEN ET AL. 2000

Chen, F.; Brown, G.; Song, M.: Overview of 3-dimensional shape measurement using optical methods. Optical Engineering 39 (2000) 1, pp. 10–22.

CHEN ET AL. 2005

Chen, H.; Xi, N.; Sheng, W.; Chen, Y.: General Framework of Optimal Tool Trajectory Planning for Free-Form Surfaces in Surface Manufacturing. Journal of Manufacturing Science and Eng. 127 (2005) 1, pp. 49–59.

CHIAVERINI ET AL. 1994

Chiaverini, S.; Siciliano, B.; Egeland, O.: Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. IEEE Transactions on Control Systems Technology 2 (1994) 2, pp. 123–134.

CONSTANTINESCU, D.; CROFT

Constantinescu, D.; Croft, E.: Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. Journal of Robotic Systems 17 (2000) 5, pp. 233–249.

COWAN 1991

Cowan, C. K.: Automatic camera and light-source placement using CAD models. Workshop on Directions in Automated CAD-Based Vision (1991), pp. 22–32.

CRAIG 2005

Craig, J.: Introduction to robotics. 3rd ed. New Jersey, USA: Pearson Prentice Hall 2005. ISBN: 978-0131236295.

CULBERSON & RECKHOW 1994

Culberson, J. C.; Reckhow, R. A.: Covering Polygons is Hard. Journal of Algorithms 1 (1994) 17, pp. 2–44.

CURLESS 1999

Curless, B.: From range scans to 3D models. ACM SIGGRAPH Computer Graphics 33 (1999) 4, pp. 38–41.

DA COSTA ET AL. 1992

Da Costa, F.; Hwang, V.; Khosla, P.; Lumia, R.: Integrated prototyping environment for programmable automation. In: Proceedings of the SPIE - The International Society for Optical Engineering, vol. 1829 of Cooperative Intelligent Robotics in Space III. Boston, MA/USA, November, 16-18, 1992, pp. 382–392.

DANNER & KAVRAKI 2000

Danner, T.; Kavraki, L.: Randomized planning for short inspection paths: In. Proceedings of the IEEE International Conference on Robotics and Automation. San Francisco, CA/USA, April 24–28, 2000, pp. 971–976.

DANTZIG ET AL. 1959

Dantzig, G. B.; Fulkerson, D. R.; Johnson, S. M.: On a Linear-Programming, Combinatorial Approach to the Traveling-Salesman Problem. Operations Research 7 (1959) 1, pp. 58-–66.

DEACON 2000

Deacon, G.: An Attempt to Raise the Level of Software Abstraction in Assembly Robotics through an Apposite Choice of Underlying Mechatronics. Journal of Intelligent & Robotic Systems 28 (2000) 4, pp. 343–399.

DENAVIT & HARTENBERG 1955

Denavit, J.; Hartenberg, R. S.: A kinematic notation for lower-pair mechanisms based on matrices. Transaction of the ASME Journal of Applied Mechanics 22 (1955), pp. 215–221.

DENKENA ET AL. 2005

Denkena, B.; Wörn, H.; Apitz, R.; Bischoff, R.; Hein, B.;Kowalski, P.; Mages, D.; Schuler, H.: Roboterprogrammierung in der Fertigung: Einfache Roboterprogrammierung für die Produktion von morgen. wt Werkstattstechnik online 95 (2005) 9, pp. 689–693.

DERIGENT ET AL. 2007

Derigent, W.; Chapotot, E.; Ris, G.; Remy, S.; Bernard, A.: 3D Digitizing Strategy Planning Approach Based on a CAD Model. Journal of Computing and Information Science in Engineering 7 (2007) 1, pp. 10–19.

DIJKSTRA 1959

Dijkstra, E.: A note on two problems in connexion with graphs. Numerische Mathematik 1 (1959), pp. 269–271.

DRESSIER ET AL. 2007

Dressier, I.; Haage, M.; Nilsson, K.; Johansson, R.; Robertsson, A.: Configuration Support and Kinematics for a Reconfigurable Gantry-Tau Manipulator. In: Proceedings of the IEEE International Conference on Robotics and Automation. Roma, Italia, April 10–14, 2007, pp. 2957–2962.

DUNN & OLAGUE 2004

Dunn, E.; Olague, G.: Multi-objective Sensor Planning for Efficient and Accurate Object Reconstruction. In: Raidl, G. et al. (Eds.): Applications of Evolutionary Computing Berlin, Heidelberg: Springer 2004, pp. 312–321.

DUTSCHKE & KEFERSTEIN 2008

Dutschke, W.; Keferstein, C.: Fertigungsmesstechnik: Praxisorientierte Grundlagen, moderne Messverfahren. 6th ed. Wiesbaden: Teubner 2008. ISBN: 978-3835101500.

EHRLENSPIEL ET AL. 2007

Ehrlenspiel, K.; Kiewert, A.; Lindemann, U.: Kostengünstig entwickeln und konstruieren. 6th ed. New York: Springer 2007. ISBN: 978-3540742227.

EICHHORN 2005

Eichhorn, A.: Optische 3D-Formerfassung für die integrierte Qualitätsprüfung von Karosserieaußenteilen. Doctoral dissertation, TU München (2005). München Hieronymus Buchreproduktions GmbH 2005. ISBN: 3-897913526.

ELLERY 2000

Ellery, A.: An introduction to space robotics. London: Springer-Praxis books in astronomy and space sciences 2000. ISBN: 978-1852331641.

ELMARAGHY & YANG 2003

ElMaraghy, H.; Yang, X.: Computer-Aided Planning of Laser Scanning of Complex Geometries. CIRP Annals - Manufacturing Technology 52 (2003) 1, pp. 411–414.

ELMARAGHY ET AL. 2009

ElMaraghy, H.; Mohib, A.; Azab, A.: Feature-based hybrid inspection planning: A mathematical programming approach. International Journal of Computer Integrated Manufacturing 22 (2009) 1, pp. 13–29.

ETTLIN 2006

Ettlin, A.: Rigid body dynamics simulation for robot motion planning. Doctoral dissertation, École polytechnique fédérale de Lausanne, Zurich (2006). <http://library.epfl.ch/theses/?nr=3663> - 02.11.2010.

EURON 2005

European Robotics Network: White paper - Industrial Robot Automation: DR-13.4 - (2005). < http://www.euron.org/miscdocs/docs/euron2/year2/dr-14-1-industry.pdf > - 11.06.2011

EUROCOM 2005

European Commission: The new SME definition - User guide and model declaration. Luxembourg: Office for Official Publications of the European Communities – (2005). ISBN: 928-9479094.

FANUC 2011

Fanuc robotics: R-30iA Controller. < http://www.fanucrobotics.de/en/Products/Controllers.aspx> -23.06.2011.

FAIZ & AGRAWAL 2000

Faiz, N.; Agrawal, S. K.: Trajectory planning of robots with dynamics and inequalities: In. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). San Francisco, CA/USA, April 24–28, 2000, pp. 3976–3982.

FEIGENBAUM 1988

Feigenbaum, A. V.: Total quality control. 3rd ed. New York: McGraw-Hill 1988. ISBN: 978-0070203549.

FENG-YUN & TIAN-SHENG 2005

Feng-yun, L.; Tian-sheng, L.: Development of a robot system for complex surfaces polishing based on CL data. The International Journal of Advanced Manufacturing Technology 26 (2005) 9, pp. 1132–1137.

FERNÁNDEZ ET AL. 2006

Fernández, P.; Álvarez, B.; Rico, J.; Blanco, D.: Constraints Evaluation and Working Methodology for Laser Scanning of Free-Form Surfaces. In: Proceeding of the IEEE Conference on Emerging Technologies and Factory Automation. Prague, Czech Republic, Sept., 20–22, 2006, pp. 523–530.

FERNÁNDEZ ET AL. 2008

Fernández, P.; Rico, J.; Álvarez, B.; Valiño, G.; Mateos, S.: Laser scan planning based on visibility analysis and space partitioning techniques. The International Journal of Advanced Manufacturing Technology 39 (2008) 7, pp. 699–715.

FIUME 1995

Fiume, E.: Isometric Piecewise Polynomial Curves. COMPUTER GRAPHICS forum 14 (1995) 1, pp. 47–58.

FRANKE & DOBROSCHKE 2010

Franke, J.; Dobroschke, A.: Robot-Based Winding-process for Flexible Coil Produciton. White paper (20110). <http://www.magnelab.com/uploads/4c 51df9035 8e8.pdf> - 02.09.2011.

FRIEDRICH 2010

Friedrich, T.: Technologieorientiertes Programmier- und Steuerungssystem für Industrieroboter. Doctoral dissertation, Technische Universität Berlin, Germany (2010). < http://opus.kobv.de/tuberlin/volltexte/2010/2822/pdf/ friedrich_thomas.pdf> - 11.06.2011.

GALANULIS 2005

Galanulis, K.: Optical Measuring Technologies in Sheet Metal Processing. Advanced Materials Research, SHEET METAL 6-8 (2005), pp. 19–34.

GATECH 2005

GATECH: Traveling Salesman Problem, Optimal Tours, Optimal USA 13509 Tour. <http://www.tsp.gatech.edu/gallery/itours/usa13509.html> - 29.07.2011.

GATZKE ET AL. 2006

Gatzke, T.; Grimm, C.; Falcidieno. B: Estimating Curvature on Triangular Meshes. International Journal of Shape Modeling 12 (2006) 1, pp. 1–29.

GELFAND 2006

Gelfand, N.: Feature Analysis and Registration of Scanned Surfaces. Doctoral dissertation, Stanford University, CA, USA (2006). <http://www2.ensc. sfu.ca/grad/theses/doctoral/yongyu_phd_ 2000.pdf> - 03.01.2011.

GERAERTS & OVERMARS 2006

Geraerts, R.; Overmars, M.: Sampling and node adding in probabilistic roadmap planners. Robotics and Autonomous Systems 54 (2006) 2, pp. 165–173.

GERMANI ET AL. 2009

Germani, M.; Mengoni, M.; Raffaeli, R.: Automation of 3D view acquisition for geometric tolerances verification: IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops) 2009, pp. 1710–1717.

GILMORE & GOMORY 1964

Gilmore, P.; Gomory, R.: Sequencing a One State-Variable Machine: A Solvable Case of the Traveling Salesman Problem. Operations Research 12 (1964) 5, pp. 655–679.

GONZALEZ-BANOS & LATOMBE 2001

Gonzalez-Banos, H.; Latombe, J.-C.: A Randomized ArtGallery Algorithm for Sensor Placement. In: Proceedings of the 17th annual symposium on Computational geometry: ACM. Medford, MA/USA, June 03–05, 2001, pp. 232–240.

GRÄSER 1999

Gräser, R.-G.: Ein Verfahren zur Kompensation temperaturinduzierter Verformungen an Industrierobotern. Doctoral dissertation, TU München (1999). München: Herbert Utz Verlag 1999. ISBN: 978-3896756039.

GROOVER 2001

Groover, P.: Automation, production systems, and computer-integrated manufacturing. 2nd ed. Upper Saddle River, New Jersey: Prentice Hall 2001. ISBN: 978-0130889782.

GROOVER 2008

Groover, M. P.: Automation, production systems, and computer-integrated manufacturing. 3rd ed. Upper Saddle River, N.J.: Prentice Hall 2008. ISBN: 978-0824703738.

HALL-HOLT 1998

Hall-Holt, O.: Scrutinizing Real-World Geometry: the Next Best View. <http://www.graphics.stanford.edu/~olaf/nbv.html> - 03.01.2011.

HEDENBORN & BOLMSJÖ 1995

Hedenborn, P.; Bolmsjö, G.: Robotics in automated inspection. International Journal of Production Economics 41 (1995) 1–3, pp. 161–166.

HEIS 2009

Kundenwissen für Forschung und Entwicklung in der Automobilindustrie: Fallstudie und Modellentwicklung zum Wissen von und über Kunden. Doctoral dissertation, Universität Augsburg (2009). München: Suedwestdeutscher Verlag fuer Hochschulschriften 2009. ISBN: 978-3838119779.

HEISEL ET AL. 1995

Heisel, U.; Richter, F.: Investigation of the Thermal Behaviour of Industrial Robots. In: Production Engineering-WGP Annals II/2 (1995), pp. 195–198.

HELLERSTEIN 2004

Hellerstein, J.; Diao, Y.; Parekh, S.; Tilbury, D.: Feedback control of computing systems. Hoboken, New Jersey: Wiley 2004. ISBN: 978-0471266372.

HESSELBACH ET AL. 2005

Hesselbach, J.; Maaß, J.; Bier, C.: Singularity Prediction for Parallel Robots for Improvement of Sensor-Integrated Assembly. CIRP Annals - Manufacturing Technology 54 (2005) 1, pp. 349–352.

HIRZINGER ET AL. 1994

Hirzinger, G.; Brunner, B.; Dietrich, J.; Heindl, J.: ROTEX - The First Remotely Controlled Robot in Space. In: Proceedings of the IEEE International Conference on Robotics and Automation. San Diego, CA/USA, May 8–13, 1994, pp. 2604–2611.

HOLLADAY 1957

Holladay, J.: A Smoothest Curve Approximation. Mathematical Tables and Other Aids to Computation 11 (1957) 60, pp. 233–243.

HOLLERBACH 1989

Hollerbach, J.: Kinematics and Dynamics for Control: Robotics science. Cambridge/MA, USA: MIT Press 1989, pp. 378–431.

HOLLMANN 2009

Hollmann, R.: Programmierzeit um rund 50% reduziert. Produktion (2009) Nr. 42, p. 15.

HU 2007

Hu, G.: Visual Servo Tracking Control via a Lyapunov-based Approach. Doctoral dissertation, University of Florida, USA (2007). <http://ncr.mae.ufl.edu/dissertations/gq.pdf > - 21.08.2011.

HUNG 2005

Hung, C.-C.: Development of an Optical Three Dimensional Measurement System and Its Performance Evaluation. Doctoral dissertation, National

Cheng Kung University, Taiwan (2005). <http://etdncku.lib.ncku.edu.tw/ETDdb/ETD- search/view_etd?URN=etd-0628105-164632> - 07.02.2010.

ISO 10303

ISO 10303-1: Industrial Automation systems and integration - Product data representation and exchange - Overview and Fundamental Principles: 1984. International Organization for Standardization (ISO).

ISO/IEC 9126-1 2001

ISO/IEC 9126-1: Software Product Evaluation -Quality Characteristics and Guidelines for their Use: 15.06.2001. International Organization for Standardization/International Electrotechnical Commission.

EDMONDS 1965

Edmonds J.: Paths, Trees, and Flowers. Canadian Journal of Mathematics 17 (1965), pp. 449–467.

JOHANSSON ET AL. 2004

Johansson, R.; Robertsson, A.; Nilsson, K.; Cederberg, P.; Olsson, M.; Olsson, T.; Bolmsjö, G.: Sensor integration in task-level programming and industrial robotic task execution control. Industrial Robot: An International Journal 31 (2004) 3, pp. 284–296.

JONSSON & OSSBAHR 2010

Jonsson, M.; Ossbahr, G.: Aspects of reconfigurable and flexible fixtures. Production Engineering - WGP Annals 4 (2010) 4, pp. 333–339.

KAPITANOVSKY & MAIMON 1993

Kapitanovsky, A.; Maimon, O.: Robot programming system for assembly: Conceptual graph-based approach. Journal of Intelligent and Robotic Systems 8 (1993) 1, pp. 35–62.

KAVRAKI ET AL. 1998

Kavraki, L. E.; Kolountzakis, M. N.; Latombe, J.-C.: Analysis of probabilistic roadmaps for path planning. IEEE Transactions on Robotics and Automation 14 (1998) 1, pp. 166–171.

KIM 2004

Kim, J. Y.: CAD-based automated robot programming in adhesive spray systems for shoe outsoles and uppers. Journal of Robotic Systems 21 (2004) 11, pp. 625–634.

KRÜGER ET AL. 2011

Krüger, J.; Schreck, G.; Surdilovic, D.: Dual arm robot for flexible and cooperative assembly. CIRP Annals - Manufacturing Technology 60 (2011) 1, pp. 5–8.

KONIETSCHKE 2007

Konietschke, R.: Planning of Workplaces with Multiple Kinematically Redundant Robots. Doctoral disertation, TU München, Germany (2007). <http://www.robotic.dlr.de/fileadmin/robotic/konietschke/Publications/dissertation_rainer_konietschke_final.pdf > - 17.07.2011.

KUGELMANN 1999

Kugelmann, D.: Aufgabenorientierte Offline-Programmierung von Industrierobotern. Doctoral dissertation, TU München (1999). München: Utz Verlag 1999. ISBN: 978-3896756152.

KUKA 2011

KUKA Roboter: KUKA KR C4. < http://www.kuka-robotics.com/germany/de/products/controllers/kr_c4/> - 23.06.2011.

KUNZMANN ET AL. 2005

Kunzmann, H.; Pfeifer, T.; Schmitt, R.; Schwenke, H.; Weckenmann, A.: Productive Metrology - Adding Value to Manufacture. CIRP Annals - Manufacturing Technology 54 (2005) 2, pp. 155–168.

LAMB ET AL.

Lamb, D. G.; Baird, D. L.; Greenspan, M. A.: An automation system for industrial 3D laser digitizing. In: Proceedings of the Second International Conference on 3-D Digital Imaging and Modeling. Ottawa, Canada, Oct. 4-8, 1999, pp. 148–157.

LARSSON & KJELLANDER J. 2008

Larsson, S.; Kjellander J.: Path planning for laser scanning with an industrial robot. Robotics and Autonomous Systems 56 (2008) 7, pp. 615–624.

LARTIGUE ET AL. 2002

Lartigue, C.; Contri, A.; Bourdet, P.: Digitised point quality in relation with point exploitation. Measurement 32 (2002) 3, pp. 193–203.

LAST & HESSELBACH 2006

Last, P.; Hesselbach, J.: A new calibration strategy for a class of parallel mechanisms. In: Lennarčič, J.; Roth, B. (Eds.): Advances in Robot Kinematics: Springer Netherlands 2006, pp. 331–338. ISBN: 978-1402049415.

LATOMBE 1991

Latombe, J.-C.: Robot Motion Planning. Norwell, MA/USA: Kluwer Academic Publishers 1991. ISBN: 0-79239206-X.

LAVALLE 2006

LaValle, S. M.: Planning algorithms. New York, NY: Cambridge University Press 2006. ISBN: 978-052186205-9.

LEE & PARK 2000

Lee, K.; Park, H.-P.: Automated inspection planning of free-form shape parts by laser scanning. Robotics and Computer-Integrated Manufacturing 16 (2000) 4, pp. 201–210.

LEFEBVRE & LAUWERS 2004

Lefebvre, P.; Lauwers, B.: STL model segmentation for multi-axis machining operations planning. Computer-Aided Design and Applications 1 (2004) 1-4, pp. 277–284.

LEVENBERG 1944

Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Quarterly Journal of Applied Mathmatics II (1944) 2, pp. 164–168.

LI & LIU 2005

Li, Y.; Liu, Z: Information entropy-based viewpoint planning for 3D object reconstruction. IEEE Transactions on Robotics 21 (2005) 3, pp. 324–337.

LI & GU 2004

Li, Y.; Gu, P.: Free-form surface inspection techniques state of the art review. Computer-Aided Design 36 (2004) 13, pp. 1395–1417.

LIN & MANOCHA 2004

Lin, M.; Manocha, D.: Collision and Proximity Queries. In: Goodman, J. et al. (Eds.): Handbook of discrete and computational geometry. Boca Raton, Florida: Chapman & Hall/CRC 2004, pp. 787–807. ISBN: 1-584883 014.

LLOYD & HAYWARD 1993

Lloyd, J.; Hayward, V.: Singularity control of robot manipulators using closed-form kinematic solutions. Canadian Conference on Electrical and Computer Engineering. Vancouver, Canada, September 14–17, 1993, pp. 1065–1068.

LORIOT ET AL. 2007

Loriot, B.; Seulin, R.; Gorria, P.; Meriaudeau, F.: Simulation for an automation of 3D acquistion and post-processing. In: Proceedings of the 8. International Conference on Quality Control by Artificial Vision (SPIE 6356). Le Creusot, France, May 23, 2007, pp. 4–10.

LOZANO-PÉREZ & WINSTON 1977

Lozano-Pérez, T.; Winston, P. H.: LAMA: A Language for Automatic Mechanical assembly. In: Proceedings of the 5th International joint Conference on Artificial Intelligence. Cambridge, MA/USA, August 22–24, 1977, pp. 710–716.

LOZANO-PEREZ 1983

Lozano-Perez, T.: Robot programming. Proceedings of the IEEE 71 (1983) 7, pp. 821–841.

LOZANO-PÉREZ & BROOKS 1986

Lozano-Pérez, T.; Brooks, R. A.: An approach to automatic robot programming: In: Proceedings of the 1986 ACM fourteenth annual conference on Computer science. Cincinnati/Ohio, USA, February 4–6, pp. 61–69.

LOZANO-PEREZ & TAYLOR 1989

Lozano-Perez, T.; Taylor, R. H.: Geometric issues in planning robot tasks: Robotics science. Cambridge, MA/USA: MIT Press 1989, pp. 227–261.

LUEDEMANN-RAVIT 2005

Luedemann-Ravit, B.: Ein System zur Automatisierung der Planung und Programmierung von industriellen Roboterapplikationen. Doctoral dissertation, Universität Dortmund, Germany (2005). <https://eldorado.tu-dortmund. de/ bitstream/2003/21662/3/Dissertation.pdf > - 16.05.2011.

LUETH 1993

Lueth, T.: Automatisierte Layoutplanung von Roboter-Fertigungszellen. Doctoral dissertation, Universität Karlsruhe, Germany (1993).

LUETH & LAENGLE 1994

Lüth, T.C.; Längle, T: Task Description, Decomposition, and Allocation in a Distributed Autonomous Robot System. In: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Munich, Germany, September 12–16, 1994, pp. 1516–1523.

MACFARLANE & CROFT 2003

Macfarlane, S.; Croft, E. A.: Jerk-bounded manipulator trajectory planning: design for real-time applications. IEEE Transactions on Robotics and Automation 19 (2003) 1, pp. 42–52. pp. 42–52.

MAHR 2003

Mahr, R.: From Zero to the Road in Seven Months: Minutes of a development record. Kunstoffe Plast Europe 93 (2003) 3, pp. 71–74.

MALAMAS ET AL. 2003

Malamas, E.; Petrakis, E.; Zervakis, M.; Petit, L.; Legat, J.-D.: A survey on industrial vision systems, applications and tools. Image and Vision Computing 21 (2003) 2, pp. 171–188.

MARSHALL & MARTIN 1993

Marshall, A.; Martin, R.: Computer vision, models and inspection. Singapore: World Scientific 1993. ISBN: 981-0207727.

MARTINS ET AL. 2005

Martins, F. A.; García-Bermejo, J. G.; Casanova, E. Z.; Perán González, J. R.: Automated 3D surface scanning based on CAD model. Mechatronics 15 (2005) 7, pp. 837–857.

MATTMÜLLER & GISLER 2009

Mattmüller, J.; Gisler, D.: Calculating a near time-optimal jerk-constrained trajectory along a specified smooth path. In: The International Journal of Advanced Manufacturing Technology 45 (2009) 9–10, pp. 1007–1016.

MEYNARD 2001

Meynard, J.-P.: Control of industrial robots through high-level task programming. Doctoral dissertation, Linköping University, Sweden (2001). <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.5347&rep=rep1&type=pdf> - 19.07.2010.

MEYER ET AL. 2002

Meyer, M.; Desbrun, M.; Schoeder, P.; Barr, A.: Discrete Differential Geometry Operators for Triangulated 2-Manifolds. In: International Workshop on Visualization and Mathematics. Berlin, Germany, May 22–25, 2002.

MILBERG ET AL. 1997

Milberg, J.; Reinhart, G.; Trunzer, W.: Strategies for Online Path Planning for Robots with 3D Contour-Following Sensors. Production Engineering - WGP Annals 4 (1997) 1, pp. 45–50.

MITSI ET AL. 2008

Mitsi, S.; Bouzakis, K.-D; Sagris, D.; Mansour, G.: Determination of optimum robot base location considering discrete end-effector positions by means of hybrid genetic algorithm. Robotics and Computer-Integrated Manufacturing 24 (2008) 1, pp. 50–59.

MITSUBISHI 2011

Mitsubishi Electric: CR robot controllers. < http://www.mitsubishi-automation.de/ products/ robots.html > - 23.06.2011.

MOTOMAN 2011

Yaskawa Motoman: DX100 robot controller < http://www.motoman.de/de/produkte/software/?no_cache=1>. - 23.06.2011.

MÖLLER ET AL. 2008

Möller, T.; Haines, E.; Hoffman, N.: Real-time rendering. 3rd ed. Wellesley, Massachusetts: A.K. Peters 2008. ISBN: 1-568814240.

MOORING ET AL. 1991

Mooring, B.; Roth, Z.; Driels, M.: Fundamentals of manipulator calibration. New York: Wiley, 1991. ISBN: 978-0471508649.

MÖNNING 2006

Mönning, F.: Effiziente robotergestützte Messtechnik durch intelligente Sensorausrichtung. Doctoral dissertation, Rheinisch-Westfaelische Technische Hochschule, Aachen: Shaker 2006. ISBN: 978-3832256937.

MUNZERT 2008

Munzert, U.: Bahnplanungsalgorithmen für das robotergestützte Remote-Laserstrahlschweißen. Doctoral dissertation, TU München (2008). Mün-

chen: Herbert Utz Verlag 2008. ISBN: 978-3831609482

MYERS 1999

Myers, D.: An approach to automated programming of industrial robots. In: Proceedings of the IEEE International Conference on Robotics and Automation. Detroit, MI/USA, Mai, 10–15, 1999, pp. 3109–3114.

NAKAMURA & HANAFUSA 1986

Nakamura, Y.; Hanafusa, H.: Inverse Kinematic Solutions With Singularity Robustness for Robot Manipulator Control. Journal of Dynamic Systems, Measurement, and Control 108 (1986) 3, pp. 163–171.

NAUMANN ET AL. 2006

Naumann, M.; Wegener, K.; Schraft, R. D.; Lachello, L.: Robot cell integration by means of application-P'n'P. Düsseldorf: VDI-Verlag 2006. ISBN: 3-180919566.

NETO ET AL. 2010A

Neto, P.; Pires, J.; Moreira, A: CAD-based off-line robot programming: In: Proceedings of the IEEE Conference on Robotics Automation and Mechatronics. Singapore, June 28–30, 2010, pp. 516–521.

NETO ET AL. 2010B

Neto, P.; Pires, N.; Moreira, P.: 3D CAD-Based Robot Programming for the SME Shop-Floor. In: Proceedings of the 20th International Conference on Flexible Automation and Intelligent Manufacturing. San Francisco, CA/ USA, July 12–14, 2010, pp. 234–239.

NEUMANN ET AL. 2009

Naumann, M.; Wegener, K.; Schraft, R. D.: Control Architecture for Robot Cells to Enable Plug'n'Produce. In: Proceedings of the IEEE International Conference on Robotics and Automation. Roma, Italy, April 10–14, 2007 pp. 287–292.

NIKU 2011

Niku, S.: Introduction to robotics. 2nd ed. Hoboken, New Jersey: Wiley 2011. ISBN: 0-470604468.

KIKON 2008

Kikon, M.: Focus Point Cloud Software - Enabling the Digital Inspection Process. Nikon Metrology (Ed.). <http://www.hoskin.qc.ca/uploadpdf/Ins

trumentation/Nikon/hoskin_focus_inspection_suite_en%5B1%5D_4c8e82ec38505.pdf> - 24.07.2011.

NNAJI 1993

Nnaji, B.: Theory of automatic robot assembly and programming. London: Chapman & Hall 1993. ISBN: 0-412393107.

NOF 1999

Nof, S.: Handbook of industrial robotics. 2nd ed. New York: Wiley 1999. ISBN: 0-471177830.

NURRE 2000

Nurre, J.: Three-Dimensional Vision. In: Shell, R. L. et al. (Eds.): Handbook of industrial automation. New York: Marcel Dekker 2000, Chapter 5.3. ISBN: 0-824703731.

OJDANIC 2009

Ojdanic, D.: Using Cartesian Space for Manipulator Motion Planning – Application in Service Robotics. Doctoral Dissertation, University of Bremen, Germany (2009). < http://www2.iat.uni-bremen.de/~amarob/paper/PhD_Ojdanic.pdf > - 29.04.2011.

OTTOVISION 2009

Otto Vision Technology GmbH: Integrierte 3D-Software für Inspektion und Flächenrückführung. <http://www.otto-jena.de/software_3d.html> - 13.08. 2011.

O'ROURKE 1987

O'Rourke, J.: Art gallery theorems and algorithms.Oxford: Oxford University Press Inc. (1987). ISBN: 0-195039653.

RENTON ET AL. 1999

Renton, P.; Greenspan, M.; Elmaraghy, H.: Plan-N-Scan: A Robotic System for Collision-Free Autonomous Exploration and Workspace Mapping. Journal of Intelligent and Robotic Systems 24 (1999) 3, pp. 207–234.

PAPADOPOULOS-ORFANOS & SCHMITT 1997

Papadopoulos-Orfanos, D.; Schmitt, F.: Automatic 3-D digitization using a laser rangefinder with a small field of view: Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling 1997, pp. 60–67.

PAULY ET AL. 2003

Pauly, M.; Keiser, R.; Gross, M.: Multi-scale feature extraction on point-sampled surfaces. COMPUTER GRAPHICS 22 (2003) 3, pp. 281–289.

PERCEPTRON 2011

Perceptron Inc. <http://www.perceptron.com/index.php/en>- 12.06.2011

PERTIN 1987

Pertin-Troccaz, J.: Online automatic robot programming: A case study in grasping: Proceedings of the IEEE International Conference on Robotics and Automation 1987, pp. 1292–1297.

PETITJEAN 2002

Petitjean, S.: A survey of methods for recovering quadrics in triangle meshes. ACM Computing Surveys (CSUR) 34 (2002) 2, pp. 211–262.

PFEIFER 2002

Pfeifer, T.: Production metrology. Oldenbourg: Oldenbourg Wissenschafts-verlag GmbH 2002. ISBN: 3-486258850

PIEPER & ROTH 1969

Pieper, D.; Roth, B.: The Kinematics of Manipulators Under Computer Con trol. In: Proceedings of the 2nd World Congress on the Theory of Machines and Mechanisms. Zakopane, Poland, Sept 23–27, 1969, pp. 159–169.

PIRES ET AL. 2004

Pires, J.; Godinho, T.; Ferreira, P.: CAD Interface for automatic robot welding programming. Industrial Robot: An Intl. Jrnl. 31 (2004) 1, pp. 71–76.

PIRES ET AL. 2005

Pires, J.; Nilsson, K.; Petersen, H. G.: Industrial robotics applications and industry-academia cooperation in Europe. IEEE Robotics & Automation Magazine 12 (2005) 3, pp. 5–6.

PILLER ET AL. 2006

Piller, F.; Koller, H.; Reichwald, R.: Mass Customization: Ein wettbewerbs-strategisches Konzept im Informationszeitalter. 4th ed. Wiesbaden: Deutsche Universitäts-Verlag 2006. ISBN: 978-3835003550.

PITO 1999

Pito, R.: A solution to the next best view problem for automated surface acquisition. IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (1999) 10, pp. 1016–1030.

POTINECKE 2010

Potinecke, T.: Methode zur Systematisierung von Teilprozessen in der Produktentwicklung beim Einsatz von CAx-Technologien. Doctoral dissertation, Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO, Universität Stuttgart, Germany(2010).

PRESSLEY 2009

Pressley, A.: Elementary Differential Geometry. 2nd ed., Guildford, Surrey: Springer London 2009. ISBN: 978-1848828902.

PRIETO ET AL. 2000

Prieto, F.; Lepage, R.; Boulanger, P.; Redarce, T.: Inspection of 3D parts using high accuracy range data. In: Proceedings oft he Machine vision applications in industrial inspection VIII. San Jose, CA/USA, January 23-28, 2000, pp. 82–93.

PRIETO ET AL. 2002

Prieto, F.; Redarce, T.; Lepage, R.; Boulanger, P.: An Automated Inspection System. The International Journal of Advanced Manufacturing Technology 19 (2002) 12, pp. 917–925.

PRIETO ET AL. 2003

Prieto, F.; Lepage, R.; Boulanger, P.; Redarce, T.: A CAD-based 3D data acquisition strategy for inspection. Machine Vision and Applications 15 (2003) 2, pp. 76–91.

PRINZ ET AL. 1996

Prinz, M.; Liu, H-C; Nnaji, B.; Lueth, T.: From CAD-based kinematic modeling to automated robot programming. In: Robotics and Computer-Integrated Manufacturing 12 (1996) 1, pp. 99–109.

PUGH 1986

Pugh, G. A.: Partitioning for selective assembly. Computers & Industrial Engineering 11 (1986) 1–4, pp. 175–179.

PURCELL 2004

Purcell, T.: Ray Tracing on a Stream Processor (Diss.): Department of Computer Science. Stanford University, CA/USA (2004). <http://graphics.stanford.edu/papers/tpurcell_thesis/tpurcell_thesis.pdf>.- 17.04.2011.

QUINLAN & KHATIB 1993

Quinlan, S.; Khatib, O.: Elastic bands: connecting path planning and control: Proceedings of the IEEE International Conference on Robotics and Automation. Atlanta, GA/USA, May –6, 1993, pp. 802–807.

RAO 2007

Rao, R.: Decision Making in the Manufacturing Environment: Using Graph Theory and Fuzzy Multiple Attribute Decision Making Methods. Berlin, Heidelberg: Springer 2007. ISBN: 978-1846288197.

RASHIDY 2009

Rashidy, H.: Knowledge-based quality control in manufacturing processes with application to the automotive industry. Doctoral dissertation, TU München (2009). München: Herbert Utz Verlag 2009. ISBN: 978-383160 8621.

REED 1998

Reed, M.: Solid model acquisition from range imagery. Doctoral dissertation, Columbia University, NY/USA (1998). < http://www.cs.columbia.edu/~allen/PAPERS/reedthesis.pdf > - 25.03.2011.

REED & ALLEN 2000

Reed, M.; Allen, P.: Constraint-based sensor planning for scene modeling. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 12, pp. 1460–1467.

REINHART ET AL. 1998

Reinhart, G.; Gräser, R.-G.; Klingel, R.: Qualification of Standard Industrial Robots to Cope with Sophisticated Assembly Tasks. CIRP Annals - Manufacturing Technology 47 (1998) 1, pp. 1–4.

REINHART ET AL. 2003

Reinhart, G.; Haag, M.; Fusch, T.; Wagner, W. In: Zäh, M. et al. (Eds.): Münchener Kolloquium: Grenzen überwinden - Wachstum der neuen Art. Munich: Utz verlag 2003, pp. 137–158. ISBN: 978-3831602223.

REINHART & RASHIDY 2008

Reinhart, G.; Rashidy, H. (2008): Further potentials of CAQ tools for fault handling in automotive manufacturing processes. Production Engineering - WGP Annals 2 (2008) 1, pp. 47–54.

REINHART ET AL. 2008

Reinhart, G.; Munzert, U.; Vogl, W.: A programming system for robot-based remote-laser-welding with conventional optics. CIRP Annals - Manufacturing Technology 57 (2008) 1, pp. 37–40.

REINHART & TEKOUO 2009

Reinhart, G.; Tekouo, W.: Automatic programming of robot-mounted 3D optical scanning devices to easily measure parts in high-variant assembly. CIRP Annals - Manufacturing Technology 58 (2009) 1, pp. 25–28.

REINHART & TEKOUO 2010

Reinhart, G.; Tekouo, W: 3D-MESSSYSTEME - Automatisierte Programmierung von Messrobotern zur Qualitätssicherung von Karosseriebauteilen. In: Zeitschrift für Wirtschaftlichen Fabrikbetrieb 105 (2010) 1, pp. 52–57.

REINHART ET AL. 2011

Reinhart, G.; Zäh, M.; Tekouo, W.; Roesel, W.; Ostgathe, M.; Lau, C.; Wiesbeck, M.; Gyger, T.; Bannat, A.; Bautze, T.; Beetz, M.; Blume, J.; Diepold, K.; Ertelt, C.; Geiger, F.; Gmeiner, T.; Knoll, A.; Lenz, C.; Roesel, W.; Ruehr, T.; Schuboe, A.; Shea, K.; Wersborg, I.; Stork, S.; Tekouo, W.; Wallhoff, F.: Artificial Cognition in Production Systems. IEEE Transactions on Automation Science and Engineering 8 (2011) 1, pp. 1–27.

REIS 2011

Reis Robotics: ROBOTstarV-PCX. < http://www.reisrobotics.de/reisrobotics/us/ROBOTS/Robot+control.html> - 23.06.2011.

RICHTER & LUETH 2010:

Richter, C.; Lueth, T.: Robot Controller Architecture for User Friendly Application Deployment. In: Proceedings of the IEEE International Conference on Robotics and Biomimetics. Tianjin, China, Dec. 14–18, 2010, pp. 484–488.

ROESSL 2005

Roessl, C.: New Techniques for the Modeling, Processing and Visualization

of Surfaces and Volumes. Doctoral dissertation, Universität des Saarlandes, Saarbrücken (2005). < http://edoc.mpg.de/get.epl?fid=22457&did=278986 &ver =0 > - 13.04.2011.

ROOKS 2001

Rooks, B.: Robots get the measure of car body inspection. Industrial Robot: An International Journal 28 (2001) 6, pp. 125–130.

ROOS & BEHRENS 1997

Roos, E.; Behrens, A.: Off-line programming of industrial robots – Adaptation of simulated user programs to the real environment. Computers in Industry 33 (1997) 1, pp. 139–150.

ROOS ET AL. 1997

Roos, E.; Behrens, A.; Anton, S.: RDS - realistic dynamic simulation of robots. In: proceedings of the 28th International Symposium on Robotics. Detroit, MI/USA, May 13–15 1997, pp. 17–27.

ROY & IQBAL 2005

Roy, A.; Iqbal, K.: PID controller tuning for the first-order-plus-dead-time process model via Hermite-Biehler theorem. ISA Transactions 44 (2005) 3, pp. 363–378.

SANSONI ET AL. 2009

Sansoni, G.; Trebeschi, M.; Docchio, F.: State-of-The-Art and Applications of 3D Imaging Sensors in Industry, Cultural Heritage, Medicine, and Criminal Investigation. Sensors - Open Access Journal 9 (2009) 1, pp. 568–601.

SCHNEEFUß & PFEIFER 2004

Schneefuß, K.; Pfeifer, T.; Optische Messtechnik -Erfolgsfaktor für die Produktion von morgen. In: Proceedings of the 24th of the Heidelberger Bildverarbeitungsforum. Stuttgart, Germany, March 9, 2004, pp. 18–22.

SCHMITT ET AL. 2011

Schmitt, R.; Pfeifer, T.; Pavim, X.: Cognitive Production Metrology - Ein neues Konzept zur qualitativen Absicherung der Kleinserienproduktion. Industrie Management 2 (2011) 27, pp. 13–18.

SCHRAFT & MEYER 2006

Schraft, R.; Meyer, C.: The need for an intuitive teaching method for small and medium enterprises: Proceedings of the Joint Conference on Robotics

37th International Symposium on Robotics and Robotik, 4th German Conference on Robotics. Munich, Germany, May 15–17, 2006, pp. 95–101.

SCHREIBER ET AL. 1999

Schreiber, G.; Otter, M.; Hirzinger, G.: Solving the singularity problem of non-redundant manipulators by constraint optimization. In: Proceedings of the International Conference on Intelligent Robots and Systems. Kyongju, South Korea, Oct. 17–21, 1999, pp. 1482–1488.

SCHREIBER 2005

Schreiber, G.: Steuerung für redundante Robotersysteme: Benutzer- und aufgabenorientierte Verwendung der Redundanz. Doctoral dissertation, Universität Stuttgart, Germany (2005). < http://elib.uni-stuttgart.de/opus/ volltexte/ 2005/ 2186/pdf/schreiberDiss.pdf > - 22.06.2011.

SCHWENKE ET AL. 2002

Schwenke, H.; Neuschäfer, U.; Pfeifer, T.; Kunzmann, H.: Optical Methods for Dimensional Metrology in Production Engineering. CIRP Annals - Manufacturing Technology 51 (2002) 2, pp. 685–699.

SCIAVICCO & SICILIANO 2005

Sciavicco, L.; Siciliano, B.: Modelling and control of robot manipulators. 2nd ed., London: Springer 2005. ISBN: 978-1852332211.

SCOTT ET AL. 2001

Scott, W.; Roth, G.; Rivest, J.: View planning with a registration constraint. In: Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling. Quebec City, Canada Mai 28–01 June, 2001, pp. 127–134.

SCOTT ET AL. 2003

Scott, W.; Roth, G.; Rivest, J: View planning for automated three-dimensional object reconstruction and inspection. ACM Computing Surveys (CSUR) 35 (2003) 1, pp. 64–96.

SCOTT 2009

Scott, W.: Model-based view planning. Machine Vision and Applications 20 (2009) 1, pp. 47–69.

SHILLER & DUBOWSKY 1991

Shiller, Z.; Dubowsky, S. (1991): On computing the global time-optimal

motions of robotic manipulators in the presence of obstacles. IEEE Transactions on Robotics and Automation, 7 (6), pp. 785–797.

SHKOLNIK 2010

Shkolnik, C.: Sample-Based Motion Planning in High-Dimensional and Dierentially-Constrained Systems. Doctoral thesis, Massachusetts Institute of Technology, MA/USA (2010). <http://groups.csail.mit.edu/robotics-center/public_papers/Shkolnik10b.pdf>- 05.09.2011.

SHU & XI 1999

Shu, C.; Xi, F.: Model-based scanning path generation for inspection. In: Proceedings of the Second International Conference on 3D Digital Imaging and Modeling. Ottawa, Canada, Oct. 4–8, 1999, pp. 118–124.

SICILIANO & KHATIB 2008

Siciliano, B.; Khatib, O. (Eds.): Springer handbook of robotics. Berlin, Heidelberg: Springer 2008. ISBN: 3-540382194.

SOLLER & HENRICH 2009

Soller, K.; Henrich, D.: Intuitive Robot Programming of Spatial Control Loops with Linear Movements. In: Kröger, T. et al. (Eds.): Advances in Robotics Research: Springer Berlin Heidelberg 2009, pp. 147–158. ISBN: 978-3642012136.

SOLOMAN 2010

Soloman, S.: Sensors and control systems in manufacturing. 2th New York: McGraw-Hill 2010. ISBN: 978-0071605724.

SALOMONS ET AL. 1993

Salomons, O., van Houten, F., Kals, H.; Review of Research in Feature-Based Design, Journal of Manufacturing Systems, 12 (1993) 2, pp. 113–132.

SOM 2010

Som, F.: Intuitives Bedieninterface für effiziente Mensch-Roboter-Kooperation. Automatisierungstechnik 58 (2010) 12, pp. 665–669.

SOMMERKORN ET AL. 2010

Sommerkorn, A.; Kubus, D.; Wahl, F.: The Basis of Control-Related Robotics Research - Open High-Rate Low-Level Control Architectures for Industrial Manipulators. In: Proceedings of the 41st International Symposium on and 2010 6th German Conference on Robotics 2010, pp. 1–8.

SON ET AL. 2003

Son, S.; Kim, S.; Lee, K.: Path planning of multi-patched freeform surfaces for laser scanning. The International Journal of Advanced Manufacturing Technology 22 (2003) 5, pp. 424–435.

SON & LEE 2003

Son, S.; Lee, K.: Automated Scan Plan Generation Using STL Meshes for 3D Stripe-Type Laser Scanner. In: Kumar, V. et al. (Eds.): Computational Science and Its Applications - ICCSA 2003: Springer Berlin/Heidelberg 2003, pp. 989–989.

SONG & SHIMADA 2009

Song, I.; Shimada, K.: Sketch-based computer-aided design tool for configuration design of automobile instrument panel. Computer-Aided Design and Applications 6 (2009) 5, pp. 585–594.

SPONG ET AL. 2006

Spong, M.; Hutchinson, S.; Vidyasagar, M.: Robot modeling and control. Hoboken, New Jersey: Wiley 2006. ISBN: 978-0471649908.

STAUBLI 2011

Stäubli: CS8 controller. < http://www.staubli.com/en/robotics/products/robot-controller/robot-controller-cs8/> - 23.06.2011.

STILMAN 2010

Stilman, M.: Global Manipulation Planning in Robot Joint Space With Task Constraints. IEEE Transactions on Robotics 26 (2010) 3, pp. 576–584.

STREPPEL ET AL. 2008

Streppel, A. H.; Klingenberg, W.; Singh, U. P.: Advances in sheet metal forming applications. International Journal of Machine Tools and Manufacture 48 (2008) 5, pp. 483–484.

SUPERVILLE C. R. & GUPTA S. 2001

Superville C.; Gupta S.: Issues in modeling, monitoring and managing quality costs. The Total Quality Management (TQM) Magazine 13 (2001) 6, pp. 419–424.

TARABANIS ET AL. 1995A

Tarabanis, K.; Allen, P.; Tsai, R.: A survey of sensor planning in computer

vision. IEEE Transactions on Robotics and Automation 11 (1995) 1, pp. 86–104.

TARABANIS ET AL. 1995B

Tarabanis, K.; Tsai, R.; Allen, P.: The MVP sensor planning system for robotic vision tasks. IEEE Transactions on Robotics and Automation 11 (1995) 1, pp. 72–85.

TARBOX & GOTTSCHLICH 1995

Tarbox, G.; Gottschlich, S.: Planning for Complete Sensor Coverage in Inspection. Computer Vision and Image Understanding 61 (1995) 1, pp. 84–111.

TEICH ET AL. 2010

Teich, T.; Militzer, J.; Jahn, F.; Neumann, T.; Kretz, D.: Step Standardized Product Data Representation and Exchange for Optimized Product Development and Automated Process Planning. In: Proceedings of the 6th CIRP sponsored International Conference on Digital Enterprise Technology. Hong Kong, Dec.14–16, 2009, pp. 41–56. ISBN: 978-3642104299.

THOMAS 2008

Thomas, U.: Automatisierte Programmierung von Robotern für Montageaufgaben. Doctoral dissertation, Technische Universität Braunschweig, Germany. Aachen: Shaker 2008. ISBN: 978-3832271015.

TORABI ET AL. 2007

Torabi, L.; Kazemi, M.; Gupta, K.: Configuration space based efficient view planning and exploration with occupancy grids: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. San Diego, CA/USA, Octo. 29–Nov. 2, 2007, pp. 2827–2832.

TRIGGS & LAUGIER 1995

Triggs, B.; Laugier, C.: Automatic camera placement for robot vision tasks. In: Proceedings of the IEEE International Conference on Robotics and Automation. Nagoya, Japan, May 22–27, 1995, pp. 1732–1737.

TSAI 1986

Tsai, M.-J.: Workspace geometric characterization and manipulability of industrial robots. Doctoral disertation, Ohio State University, USA (1986). http://rave.ohiolink.edu/etdc/view?acc_num=osu1260297835 - 12.05.2011.

TSIANOS ET AL. 2007

Tsianos, K.; Sucan, I.; Kavraki, L.: Sampling-based robot motion planning: Towards realistic applicat. Computer Science Review 1 (2007) 1, pp. 2–11.

UHLMANN ET AL. 2008

Uhlmann, E.; Friedrich, T.; Bayat, N.: Development of a technology orientated programming system for industrial robots. Production Engineering - WGP Annals, 2 (2008) 1, pp. 103–108.

URRUTIA 2000

Urrutia, J.: Art Gallery and Illumination Problems. In: Sack, J.-R. et al. (Eds.): Handbook of computational geometry. Amsterdam, New York: Elsevier 2000, pp. 973–1027. ISBN: 978-0444825377.

VANDENBRANDE & REQUICHA 1993

Vandenbrande, J. H.; Requicha, A.: Spatial Reasoning for the Automatic Recognition of Machinable Features in Solid Models. IEEE Transactions on Pattern Analysis and Machine Intelligence 15 (1993) 12, pp. 1269-1285.

VAN DIJK 2007

van Dijk, N.; van de Wouw, N.; Pancras, W.; Nijmeijer, H.: Time-optimal trajectory generation for industrial manipulators along predefined paths with kinematic constraints. In: Proceeding of the 26th Benelux Meeting on Systems and Control. Lommel, Belgium, March 13–15, 2007, pp. 178–184.

VERL ET AL. 2008

Verl, A.; Boye, T.; Pott, A.: Measurement pose selection and calibration forecast for manipulators with complex kinematic structures. CIRP Annals - Manufacturing Technology 57 (2008) 1, pp. 425–428.

VERL 2009

Verl, A.: Patent US 2009/0099690 A1. Method for robot-assisted measurement of measurable objects. Filed: may 6, 2005, issued April 16, 2009. Assignee: KUKA Roboter GmbH.

VERL ET AL. 2010

Verl, A.; Schmitz, S.; Yang, D.; Wurst, K-H: Industrial powerline communication for machine tools and robotics. Production Engineering - WGP Annals 4 (2010) 2–3, pp. 295–305.

VERSCHEURE ET AL. 2009

Verscheure, D.; Demeulenaere, B.; Swevers, J.; Schutter, J. de; Diehl, M.: Time-Optimal Path Tracking for Robots: A Convex Optimization Approach. IEEE Transactions on Automatic Control, 54 (2009) 10, pp. 2318–2327.

VOGL 2008

Vogl, W.: Eine interaktive räumliche Benutzerschnittstelle für die Programmierung von Industrierobotern. Doctoral dissertation, TU München (2008). München: Herbert Utz Verlag 2008. ISBN: 978-3831608690.

WAMPLER 1986

Wampler, C. W.: Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods. IEEE Transactions on Systems, Man and Cybernetics 16 (1986) 1, pp. 93–101.

WECK & BRECHER 2005

Weck, M.; Brecher, C.: Werkzeugmaschinen - Maschinenarten und Anwendungsbereiche. 6th ed., Heidelberg: Springer 2005. ISBN: 3-540225048.

WECKENMANN ET AL. 2001

Weckenmann, A.; Knauer, M.; Killmaier, T.: Uncertainty of coordinate measurements on sheet-metal parts in the automotive industry. Journal of Materials Processing Technology 115 (2001) 1, pp. 9–13.

WECKENMANN & GABBIA 2006

Weckenmann, A.; Gabbia, A.: Testing formed sheet metal parts using fringe projection and evaluation by virtual distortion. In: Osten, W. (Eds.): Fringe 2005. Berlin, Heidelberg: Springer-Verlag 2006, pp. 539-546. ISBN: 978-3540260370.

WECKENMANN ET AL. 2009

Weckenmann, A.; Jiang, X.; Sommer, K.; Neuschaefer-Rube, J.; Shaw, U.; Seewig, L.; Estler, T.: Multisensor data fusion in dimensional metrolo-gy. CIRP Annals - Manufacturing Technology 58 (2009) 2, pp. 701–721.

WESTKÄMPER ET AL. 1998

Westkämper, E.; Schraft, R.; Schweizer, M.; Fred H., T.; Meißner, A.: Task-oriented programming of large redundant robot motion. Robotics and Computer-Integrated Manufacturing 14 (1998) 5–6, pp. 363–375.

WIERDA 1991

Wierda, L.: Linking Design, Process Planning and Cost Information by Feature-Based Modeling. Journal of Engineering Design 2 (1991) 1, pp 3–19.

WANG & OZSOY 1991

Wang, N.; Ozsoy, T.: A scheme to represent features, dimensions, and tolerances in geometric modeling. Journal of Manufacturing Systems 10 (1991) 3, pp. 233–240.

WONG & KAMEL 2004

Wong, C.; Kamel, M.: Comparing viewpoint evaluation functions for model-based inspectional coverage. Proceedings of the 1st Canadian Conference on Computer and Robot Vision. Ontario, Canada, may 17-19, 2004, pp. 287–294.

WOOD 1994

Wood, S.: An Architecture for a function friven mechanical design solution library. Doctoral dissertation, Oregon State University, USA (2006). <http://my.fit.edu/~swood/PhD%20Dissertation%201994.pdf> -23.07.2011.

WU ET AL. 2009

Wu, C.; Mao, X.; Shi, X.; Zhang, L.: Methods of Generating Robot Spraying Trajectory Based on Shoe Sole Information: International Workshop on Intelligent Systems and Applications 2009, pp. 1–4.

WULFSBERG & CLAUSING 2008

Wulfsberg, J; Clausing, N.: Locally Flexible Measuring Robot for Quality Control. In: Proceedings of the 2th CIRP Conference on Assembly Technologies and Systems CATS. Toronto, Canada, 21–23 Sept. 2008, pp. 59–68.

ZÄH & RASHIDY 2006

Zäh, M.; Rashidy, H.: Automatisierte Qualitätsregelkreise im Karosseriebau - Tiefer greifende Automatisierung für die Qualitätstechnik von morgen . wt Werkstattstechnik online 96 (2006) 9, pp. 597–601.

ZÄH & PRASCH 2007

Zäh, M.; Prasch, M.: Systematic Workplace and Assembly Redesign for Aging Workforces. Production Engineering - WGP Annals 1 (2007) 1, pp. 57–64.

11 List of Figures

List of Figures

12 List of Tables

List of Tables

Appendix A

Table A.1: DH-parameter of the manipulator integrated in the experimental platform

$\boldsymbol{Link\ (i)}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{a_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$
1	90°	260	675	θ_1
2	0	680	0	θ_2
3	90°	-35	0	θ_3
4	-90°	0	-670	θ_4
5	90°	0	0	θ_5
6	0	0	-158	θ_6

Table A.2: Geometrical parameters of the light section sensor mounted on the manipulator in the experimental Platform

Parameter	Numerical values	Unit
Max View Angle (θ_{lim})	45	°
Depth of field (DoFi)	170-230	mm
beam with (LaWi)	60	mm
Focus length	200	mm
Number of laser rays (n_{LS})	10	-
Aperture angle (α_{LS})	30	°