Ein holistisches Konzept zur Entwicklung einer innovativen, umfassenden und multisensoriellen Steuerung für Nanopositionier- und Nanomessmaschinen

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur
(Dr.-Ing.)

vorgelegt der
Fakultät für Maschinenbau der
Technischen Universität Ilmenau

von Herrn

Dipl.-Ing. Erik Sparrer

geboren am 02.07.1981 in Erlabrunn/Deutschland

urn:nbn:de:gbv:ilm1-2015000516

Kurzfassung

Die vorliegende Arbeit ist ein Beitrag zur Verknüpfung von etablierten Sensor-Aktor-Lösungen der Nanometrologie mit Interaktionskonzepten aus dem Gebiet der makroskopischen Koordinatenmesstechnik. Viele Teilaspekte beider Disziplinen weisen methodische Analogien auf und legen die synergetische Nutzung einzelner Ansätze der Koodinatenmesstechnik im Rahmen nanometrologischer Anwendungen nahe. Durch eine mehrschichtige Abstraktion wird die transparente Steuerung von Nanopositionier- und Nanomessmaschinen ermöglicht, wodurch neuartige und zukunftsweisende Bedienkonzepte für die Nanometrologie entstehen. Insbesondere soll die Frage beantwortet werden, ob und wie die Ansätze zur Steuerung makroskopischer Messsysteme für die Nutzung mit dem funktionell ähnlichen, nanometrologischen System Nanopositionier- und Nanomessmaschinen auf Basis der allgemein anerkannten Steuerungsschnittstelle *Inspection plus plus for Dimensional Measurement Equipment* (I++DME) ermöglicht werden können.

Die grundlegenden Steuerungsansätze und Verfahrensweisen zur Steuerung bzw. Messung von und mit makroskopischen bzw. nanometrologischen Systemen werden vorgestellt und erörtert. Insbesondere werden dabei die Bedeutung und Funktion der Steuerungsschnittstelle I++DME sowie einer Erweiterung für optische Sensoren, der *Optical Sensor Interface Standard* (OSIS), behandelt. Anschließend wird die Eignung der Schnittstellen für den Betrieb von Nanopositionier- und Nanomessmaschinen umfassend auf funktionaler und informationstechnischer Ebene untersucht. Aufbauend auf diesen Erkenntnissen wird ein frei skalierbares, auf mehrstufige Schnittstellenschichten gestütztes Konzept für eine Steuerungssoftware für Nanopositionier- und Nanomessmaschinen, zusätzliche messtechnische Apparaturen sowie taktile und optische Sensoren entworfen. Alle Aspekte zur Integration von Teilsystemen werden umfassend analysiert und modular in das Lösungskonzept eingebunden. Das so entstandene innovative holistische Konzept zur transparenten Steuerung von nanometrologischen multisensoriellen Messungen mit Nanopositionier- und Nanomessmaschinen wird im weiteren Verlauf der Arbeit sukzessive ausgebaut und zu einem funktionstüchtigen Demonstrator integriert.

Mit umfangreichen Analysen werden das Laufzeitverhalten, sowie die messtechnischen Eigenschaften der entstandenen Schnittstellensoftware ermittelt und mit metrologischen Experimenten bewertet. Einen zentralen Aspekt stellt die Interoperabilität und Unabhängigkeit der implementierten Steuerung von Anwendersoftware dar. Zur Demonstration der erfolgreichen Abstraktion des Systems Nanopositionier- und Nanomessmaschine als generisches Messsystem, werden verschiedene kommerzielle Steuerungsprogramme für makroskopische Koordinatenmessgeräte mit dem Demonstrator erprobt. Die im Verlauf dieser Experimente gewonnenen Erkenntnisse über den Arbeitsablauf zum Durchführen einer Messung und die Eignung der Programme für nanometrologische Anwendungen werden abschließend zusammengefasst und diskutiert.

Abstract

The present work is a contribution to link the well-established nano-metrological sensor-actuator solutions with interaction concepts from the field of macroscopic coordinate metrology. Many aspects of both disciplines have methodological analogies and suggest the synergetic use of individual approaches of the macroscopic coordinate metrology for nano-metrological applications. Through the use of synergies the control of nano-positioning and nano-measuring machines is abstracted, whereby novel and pioneering operating concepts are created. In particular, the research work focuses on the question of whether and how the well-evolved controlling approaches for macroscopic dimensional measuring systems can be utilized with functionally similar, nano-metrology systems like nano-positioning and nano-measuring machines based on the generally accepted control interface *Inspection plus plus for Dimensional Measurement Equipment* (I ++ DME).

The basic control concepts and procedures for measurement with macroscopic or nano-metrology systems are discussed. In particular, the importance and function of the control interface I++DME as well as an extension for optical sensors, the *Optical Sensor Interface Standard* (OSIS), are treated. Profound analyzes examine the suitability of the interfaces for the operation of nano-positioning and nano-measuring machine at the functional and information technological level. Up on these findings an adaptive, multi-layer interface concept for a control software is designed, covering nano-positioning and nano-measuring machines, additional metrological equipment, as well as tactile and optical sensors. All aspects regarding the integration of sub-systems are comprehensive analyzed and integrated into the modular solution concept. The resulting innovative holistic approach to transparently control nano-metrology multi-sensor measurements with nano-positioning and nano-measuring machines is gradually expanded in the course of the work and integrated into a functional demonstrator.

The run-time behavior and metrological properties of the resulting interface software are extensively analyzed and evaluated with metrological experiments. As interoperability and universality to user software are central aspects of the implementation, the successful abstraction of a nano-positioning and nano-measuring machine as a generic metrology system is demonstrated with various commercial control programs for macroscopic coordinate measuring machines by means of the demonstrator. At last experimentally gained insight on the utilization workflow and the suitability of the control programs are summarized and discussed in a separate section.

Inhalt

Abbildungen

Tabellen

Abkürzungen

ASCII	American Standard Code for Information Interchange
CAD	Computer Aided Design
COM	Component Object Model
CORBA	Common Object Request Broker Architecture
CS	Control System – Subsystem der NPM-Maschine 200 für die (Positions-) Regelung und Bahnplanung
DAS	Data Acquisition System – Subsystem der NPMM-200 für die Messdatenerfassung und -Vorverarbeitung
DCOM	Distributed Component Object Model
DLL	Dynamic Link Library
DME	Dimensional Measuring Equipment
DMIS	Dimensional Measurement Interface Standard
DSI	Dynamic Skeleton Interface
GUI	Graphical User Interface
I++DME	Inspection plus plus for Dimensional Measuring Equipment
IDL	Interface Definition Language
IP	Internet Protocol
KMG	Koordinatenmessgerät
NIST	National Institute for Standards and Technology
NPMM, NPM-Maschine	Nanopositionier- und Nanomessmaschine
ORB	Object Request Broker
OSIS	Optical Sensor Interface Standard
RMI	Remote Method Invocation
RPC	Remote Procedure Call
SCS	Sequence Control System – Subsystem der NPMM-200 für die Ablaufsteuerung
TCP	Transmission Control Protocol
UML	Unified Modeling Language
USB	Universal Serial Bus
WG	Work Group
WinAPI	Windows Application Programming Interface
WLI	Weißlichtinterferometrie

Formelzeichen

Lateinische Formelzeichen

a_{max}	Parameter für den Bahngenerator der NPM-Maschine: maximal erlaubte Beschleunigung
α_ϑ	thermischer Ausdehnungskoeffizient
b	Basisabstand (Abstand zwischen zwei Stereokameras)
B	Bandbreite einer Lichtquelle
C	Kovarianzmatrix
$d(\vec{m})$	Mahalanobisabstand eines Parametervektors zum Schwerpunkt der Parameterwolke
f	Brennweite
F_{mess}	Antast- oder Messkraft (währen einer taktilen Antastung zwischen Taster und Objekt wirkend)
f_{mess}	Messfrequenz der NPMM-200
f_{ctrl}	Regelzungsfrequenz der NPMM-200
$\mathcal{G}$	Menge der ganzen Zahlen
$I(x,y,z)$	lokale durch (Weißlicht-) Interferenz hervorgefufene Intensität auf einem Messobjekt
$\vec{I}$	Messwertvektor eines, von einem Pixel beobachteten Interferenzverlaufes auf der Objektoberfläche
$\tilde{I}(k)$	Fouriertransformierte von $\vec{I_l}(z)$
$I_i(z)$	Interferenzverlauf, welcher an einem festen Ort auf der Objektoberfläche zu beobachten ist
j_{max}	maximal erlaubter Ruck (Parameter der Bahngenerators der NPM-Maschine - engl. jerk)
k_i	Koeffizienten für das radiale Verzeichnungsmodell einer Lochkamera
$\bar{L}$	mittlere Latenz
l_c	Kohärenzlänge einer Lichtquelle
l_{mess}	Länge des Messarmes eines Michelsoninterferometers
l_{ref}	Länge des Referenzarmes eines Michelsoninterferometers
$\overrightarrow{m_{opt}}$	das Spektrum der Lichtquelle am besten beschreibender Merkmalsvektor
$\vec{m}$	an einem Pixel ermittelter Merkmalsvektor mit Parametern für das Modell des Lichtquellenspektrums
$\bar{m}$	Schwerpunkt der Parameterwolke aller $\vec{m}$
p_i	Koeffizienten für das tangentiale Verzeichnungsmodell einer Lochkamera
Q	Kameramatrix
R	Rotationsmatrix
$s(\tilde{I})$	Modell eines Lichtquellenspektrums
t	Translationsvektor
$v_{Approach}$	Geschwindigkeit mit der Taster für eine Messung von einem Punkt ohne Kontakt zum Messobjekt zum Objekt hin bewegt wird
$v_{Retract}$	Geschwindigkeit mit welcher der Taster nach der Antastung der Oberfläche vom Objekt weg bewegt wird
v_{Search}	Geschwindigkeit mit welcher Taster und Objekt aufeinander zubewegt werden um nach der Objektoberfläche zu suchen
$\vec{X}$	Punkt im dreidimensionalen Raum

$\vec{x}$	Punkt auf der Sensorebene (im zweidimensionalen Raum)

Griechische Formelzeichen

Δl	eine Längendifferenz
Δz	Tiefenauflösung
ε_{abs}	absolute Auflösung
ε_m	Messunsicherheit
ε_{rel}	relative Auflösung
ε_t	Toleranz
λ	Wellenlänge
λ_0	dominante Wellenlänge des Lichtquellenspektrums
σ	Standardabweichung eines gaußschen Spektralmodells

Glossar

ableiten	(siehe vererben)
Compiler	(engl. Übersetzer) der Compiler übersetzt den vom Programmierer geschriebenen, vom Menschen lesbaren Quelltext, zu einem semantisch identischen jedoch vom Prozessor lesbaren Programm.
Destruktor	der Destruktor ist eine Klassenmethode welche beim Vernichten eines Objekts automatisch aufgerufen wird.
Eigenschaft	Eigenschaften sind innerhalb einer Klasse zugängliche Variable. Verschiedene Objekte einer Klasse besitzen eigene, von anderen Objekten unabhängige, Eigenschaften.
Framework	(engl. Rahmengerüst) Ein Framework ist ein Programmiergerüst, welche für die Lösung von Programmieraufgaben verwendet werden kann. Das F. erleichtert die Programmierung von Anwendungssoftware indem es dem Programmierer häufig auftretende Standardaufgaben abnimmt oder deren Bearbeitung erleichtert.
geschützt	bezeichnet ein Zugriffsrecht für Methoden und Eigenschaften einer Klasse. Geschützte Elemente sind vererbbar, jedoch nur innerhalb der Klasse zugänglich.
Heap	(engl. Haldenspeicher) Der Heap ist ein Speicherbereich ohne automatische Speicherverwaltung. Heapspeicher z. B. für eine Variable kann mit dem `new` Operator angefordert werden. Variablen oder Objekte welche auf dem Heap angelegt wurden können nur über einen Zeiger angesprochen werden. Heapspeicher muss vom Nutzer wieder freigegeben werden.
Instanz	(siehe Objekt)
instanziieren	Ein Objekt zu instanziieren bedeutet, von der entsprechenden Klasse eine Realisierung zu erzeugen. Erst die Instanz einer Klasse kann in einem Programm genutzt werden.
Klasse	Eine Klasse ist die grundlegende Kapselungseinheit der objektorientieren Programmierung. Eine Klasse kapselt Eigenschaften und Methoden und kann mit einem Bauplan gleichgesetzt werden. Üblicherweise müssen von Klassen Objekte instanziiert werden, welche im Rahmen eines Programms genutzt werden können.
Konstruktor	Der Konstruktor ist eine Klassenmethode welche automatisch beim Erstellen eines Objekts aufgerufen wird.
Methode	Methoden sind fest zu einer Klasse gehörende Funktionen.
Objekt	Ein Objekt ist die Realisierung oder Ausprägung einer Klasse. Als Analogon: Die Klasse sei *Mensch*, so ist *Isaac Newton* ein Objekt dieser Klasse.

Operator — Bestimmte nicht alphanumerische Zeichen(-ketten) stellen in Programmiersprachen Operatoren dar. Die Operatoren dienen in der Regel der Kapselung grundlegender mathematischer Funktionen (z. B. Addition). Eine Ausnahme bilden die Operatoren `new` und `delete`. Sie dienen der Allokation bzw. Deallokation von Heapspeicher (vgl. Heap) für ein Programm.

öffentlich — bezeichnet ein Zugriffsrecht für Methoden und Eigenschaften einer Klasse. Öffentliche Elemente sind vererbbar und frei zugänglich.

privat — bezeichnet ein Zugriffsrecht für Methoden und Eigenschaften einer Klasse. Private Elemente können nicht vererbt werden und sind nur innerhalb der Klasse zugänglich.

Schnittstelle — (auch abstrakte Schnittstelle) eine Schnittstelle ist eine besondere Klasse, welche Methoden und Eigenschaften definiert, jedoch selbst keine Objekte ausprägen darf.

Stack — (engl. Stapelspeicher) Der Stack ist ein Speicherbereich vergleichbar dem Heap, im Gegensatz zu diesem wird dieser Speicher jedoch automatisch verwaltet. Bei einem Funktionsaufruf reserviert der Compiler Speicher für Aufruf- und Rückgabeparameter und alle Variablen und Objekte, welche nicht mit dem `new` Operator erstellt wurden. Beim Verlassen der Funktion wird der Stapelspeicher automatisch wieder freigegeben.

transparent — Ist ein System transparent, ist es für den Anwender im normalen Betrieb nicht wahrnehmbar. Seitens des Anwenders ist kein Wissen über die Existent und damit auch die Funktion des transparenten Systems erforderlich, um einen Dienst in vollem Umfang nutzen zu können.

überladen — Überladung tritt nur in Verbindung mit Vererbung auf. Durch überladen können Methoden oder Attribute von einer Basisklasse durch die erbende Klasse verändert werden. So ist es möglich in gleichabgeleiteten Klassen verschiedene Datenverarbeitungsalgorithmen in derselben Methode zu implementieren.

(ver-)erben — Vererbung ist ein Konzept der objektorientierten Programmierung. Durch vererben können der abgeleiteten Klasse Elemente (Methoden und Eigenschaften) hinzugefügt oder Elemente der Basisklasse verändert werden.

x86-Prozessor — Ist eine weit verbreitete Prozessorfamilie mit einem gemeinsamen Befehlssatzes. Die Architektur geht auf den 1978 eingeführten 8086-Prozessor von Intel zurück.

Zeiger — Der Zeiger ist die Adresse einer Speicherzelle des Arbeitsspeichers. In der Regel verweist ein Zeiger auf den Beginn eines zusammenhängenden Speicherbereiches für Variablen oder Objekte.

Kapitel 1 Einführung

In allen produzierenden Industriezweigen stellt die Fertigungsqualität eines der wichtigsten Gütekriterien für ein Produkt dar. Historisch gesehen entstand so ein äußerst dynamischer Prozess, welcher dazu führte, dass kontinuierlich neue Fertigungsmethoden mit wachsender Güte entwickelt wurden, wodurch die Herstellung von Objekten mit hoher Gestalttreue und kleineren Strukturausdehnungen ermöglicht wurde. Insbesondere wurde dieser Trend durch die Halbleiterindustrie beflügelt, welche mit der *International Technology Roadmap for Semiconductors* (kurz ITRS) eine Komplexitätssteigerung integrierter Schaltkreise etwa alle 2,5 Jahre [1] festlegt. Diese als Mooresches Gesetz bezeichnete Leitlinie erweist sich seit nunmehr über drei Dekaden als zutreffend und wird aller Voraussicht nach seine Gültigkeit auch im nächsten Jahrzehnt behalten. Die gegenwärtige Zielstellung der ITRS sieht eine Verkleinerung der Ausdehnungen von DRAM-Zellen von derzeit 40 nm auf 12,5 nm bis 2021 vor und auch ab dem Jahr 2021 ist vom heutigen Standpunkt kein Ende des Trends absehbar. Eine gegenwärtige Prognose nach Gelsinger [2] verleiht dem Mooreschen Gesetz bis wenigstens 2029 Geltung, was nach heutigem Standpunkt Strukturen mit einer Ausdehnung von 5 nm entspräche. Doch auch mit dem Erreichen von 5 nm Strukturen ist die Grenze der technologischen Möglichkeiten nicht erschöpft. So konnte Fuechsle [3] demonstrieren, dass mit geeigneten Verfahren Transistoren auf Basis einzelner Halbleiteratome hergestellt werden können.

Neben der Halbleiterindustrie dringt jedoch auch die Herstellung von Präzisionssystemen optischer [4] und mikromechanischer [5] Bauteile in neue, kleinere Dimensionen vor. Im Rahmen der Qualitätssicherung werden für diese miniaturisierten und zum Teil komplex strukturierten Bauteile metrologische Daten zur Steuerung des Fertigungsprozesses benötigt, womit völlig neue

Herausforderungen an die Messtechnik entstehen. Die Objekte verbinden oft Makrogeometrien mit einer mikro- bzw. nanostrukturierten Oberfläche mit Steigungswinkeln von bis zu 80°, wie im Fall der mikrostrukturierten Linse des ITO Stuttgart (Abbildung 1) oder des photonischen Kristalls des Karlsruher Instituts für Technologie (Abbildung 2). Diese neuen Fertigungsverfahren erfordern es, reproduzierbar und nanometergenau über Bereiche von mehreren Millimetern zu messen.

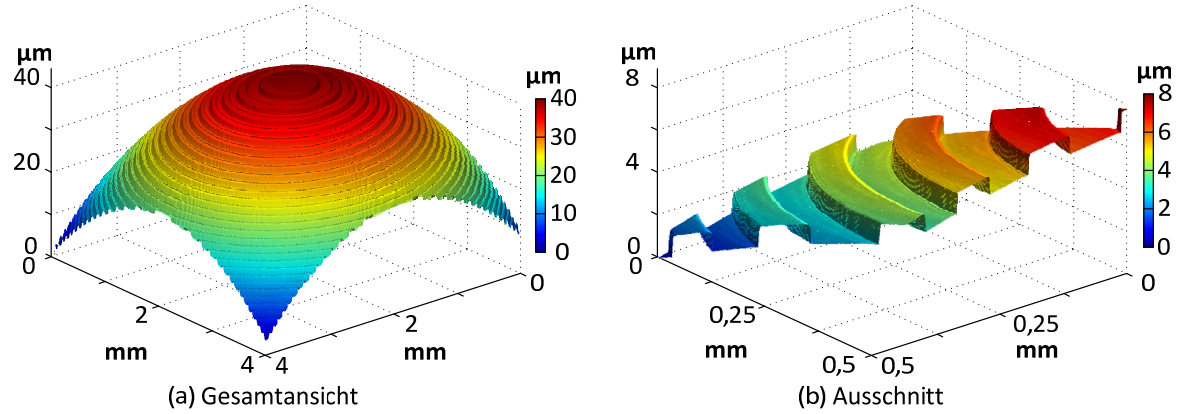


Abbildung 1: Oberfläche einer mikrostrukturierten Linse, hergestellt am IOT Stuttgart [6]

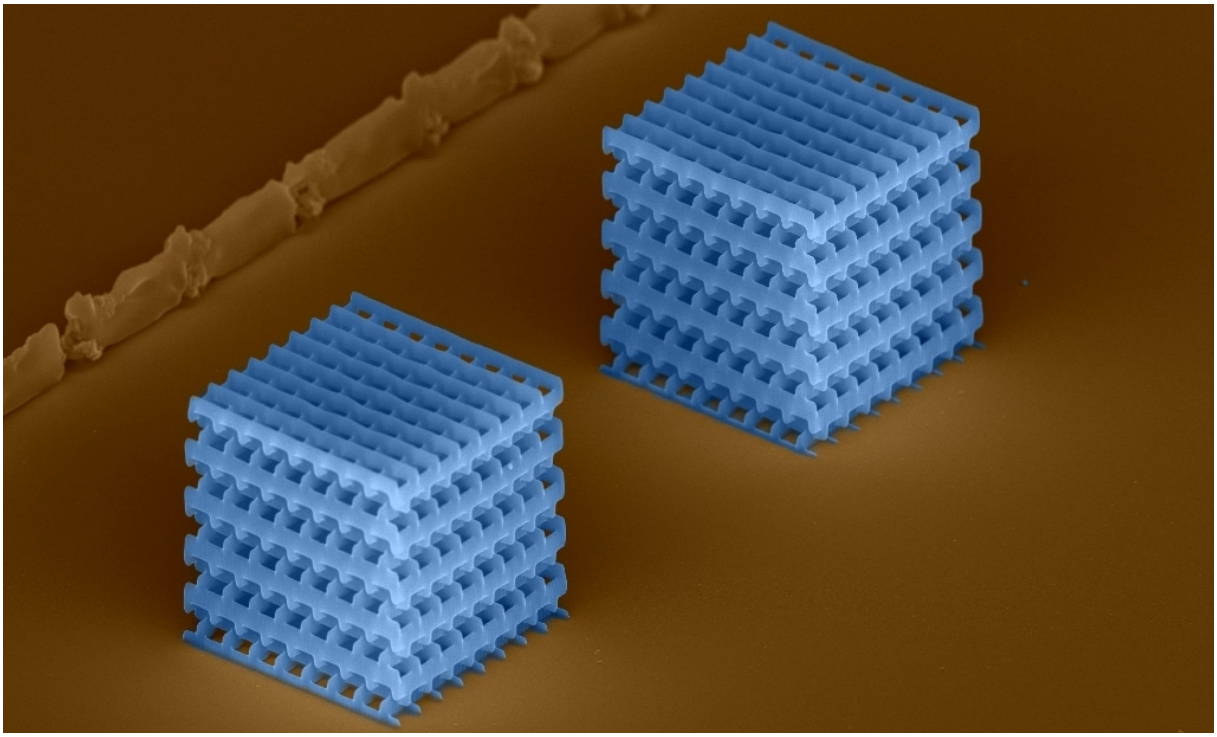


Abbildung 2: Photonische Kristalle, gefertigt am Karlsruher Institut für Technologie [7]

Neue, hochpräzise Messmethoden und -geräte müssen entwickelt werden, wodurch ein hoher Innovationsdruck auf die Metrologie entsteht. Seit 2006 sind diese Anforderungen konkret in der iMERA (*implementing Metrology in the European Research Area)*, der Metrologie-Roadmap für die Micro- und Nanometrologie [8] festgehalten. Vergleichbar zur ITRS stellt die iMERA ein zeitliche Leitlinie mit Anforderungen und Meilensteinen zur Entwicklung von dimensionellen Präzisionssmesssystemen dar (siehe Abbildung 3).

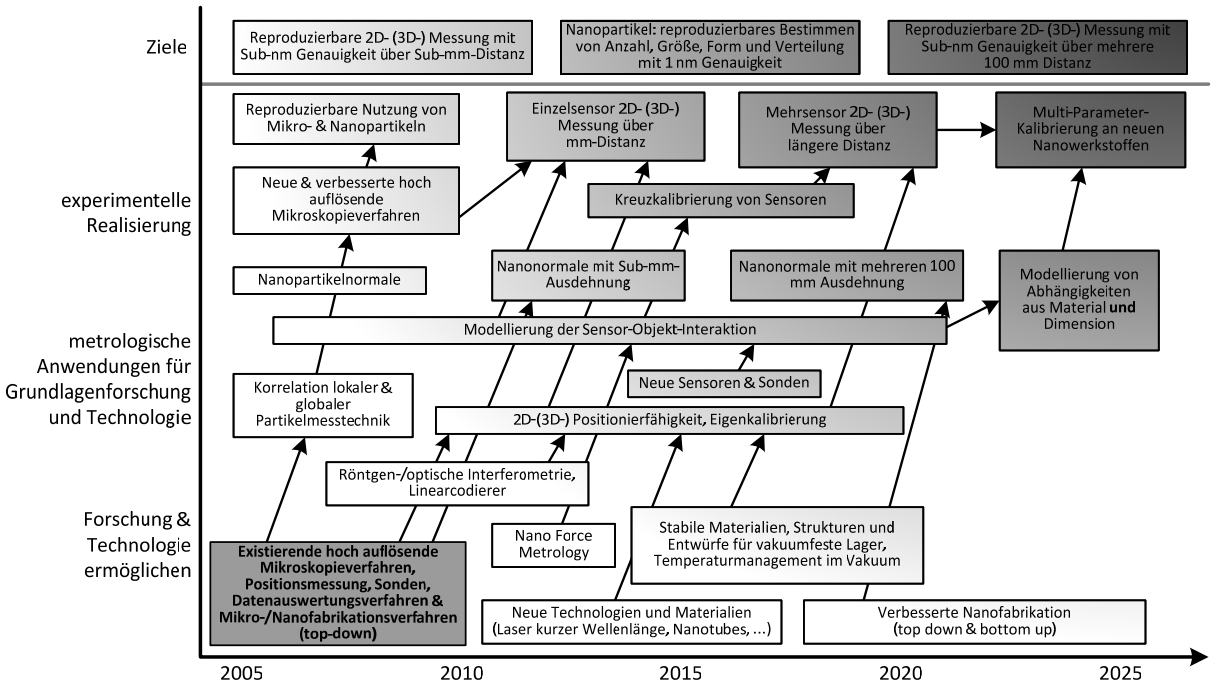


Abbildung 3: Roadmap für Mikro und Nanomesstechnik [9], [10]

Für Messanwendungen wie Präzisionsoberflächen, gekrümmte Flächen, Freiformgeometrien sowie ausgedehnte überlagerte dreidimensionale Makro- und Nanogestalten oder Strukturen mit hohen Aspektverhältnissen wird beabsichtigt, bis 2025 nanometrologische Messsysteme mit folgenden Eigenschaften [10] zu entwickeln:

- zwei- und dreidimensional messend
- Subnanometer-Reproduzierbkeit
- Stellbereiche von mehreren 100 mm je Achse
- Messen mit unterschiedlichen optischen und taktilen Sensoren
- Messdatenfusion für alle verwendeten Sensoren
- Vakuumtauglichkeit
- Subnanometerpräzision

Da die Anforderungen der iMERA die derzeitigen technischen Grenzen weit überschreiten, stellt die Umsetzung dieser Leitlinie Forscher und Ingenieure weltweit vor neue Herausforderungen. Die einfache Skalierung makroskopischer Verfahren und Systeme ist nur begrenzt möglich und insbesondere mit Hinblick auf die Realisierung von Präzisionsmessgeräten mit einer angestrebten Reproduzierbarkeit im (Sub-)Nanometerbereich vollkommen ungeeignet. Vielmehr müssen neuartige, ganzheitliche Ansätze entwickelt werden, welche den Aufbau und die Funktionsweise konventioneller metrologischer Systeme im Sinne der Nanometrologie neu bewerten. Innovative Verfahren und Strategien gepaart mit entsprechend hochgenauen Systemen sind erforderlich, um das Ziel der umfassenden Minimierung von Störeinflüssen zu gewährleisten [9].

Einen entscheidenden Beitrag zur Lösung dieser Fragestellungen leistete dabei der Sonderforschungsbereich 622 "Nanopositionier- und Nanomessmaschine". An der Technischen Universität Ilmenau arbeitete zwischen 2002 und 2013 ein interdisziplinäres Forscherteam am Aufbau neuer, tragfähiger metrologischer Konzepte für eine messtechnische Basis auf der Nanometerskala. Die Forschungsergebnisse flossen zu großen Teilen in die Entwicklung hochpräziser Nanopositionier- und Nanomessmaschinen (NPM-Maschinen) und daran angelehnter metrologischer Systeme zur funktionellen Erweiterung ein. Dadurch ist ein äußerst komplexes, hochpräzises und universell

einsetzbares Positioniersystem entstanden, welches mit verschiedenen Sensoren für metrologische Analysen im Bereich zwischen Makro- und Nanomesterskala angewendet werden kann.

Die hohe Leistungsfähigkeit moderner NPM-Maschinen und ihr kontinuierlich gewachsener Funktionsumfang rückt die Bedeutung von Software zur Steuerung, Überwachung und Verwaltung der vielfältigen und heterogenen Teilsysteme bei der Integration nanometrologischer Koordinatenmessgeräte zunehmend in den Vordergrund. Gemäß der Nano-Metrologie-Roadmap sollen diese Systeme durch eine flexible Steuerungssoftware zu einer iMERA-konformen NPM-Maschine erweitert werden. Diese Systeme sollen umfangreiche Positionier- und Messfunktionen in einem Messvolumen von mehreren Dezimetern je Achse bereitstellen, um Prüflinge automatisiert mit taktilen bzw. optischen Sensoren vermessen zu können und darüber hinaus mit Manipulatoren zu bearbeiten.

Insbesondere für die Berücksichtigung der realisierungs- bzw. messprinzipbedingten Anforderungen von Teilsystemen existiert gegenwärtig kein geschlossenes Konzept, um alle metrologischen Teilsysteme (Positioniersystem, Rotationsachsen, Sensoren und Sensorwechseleinrichtung) einer NPM-Maschine konsistent zu einem funktionsfähigen Gesamtsystem zusammenzufügen. Somit ist es auch nicht möglich, die Funktionalität derartiger Teilsysteme sinnvoll mit dem Positioniersystem NPM-Maschine zu kombinieren, wodurch das umfangreiche Potential dieses hochkomplexen und äußerst vielseitigen metrologischen Präzisionssystems nicht erschöpfend genutzt werden kann. Die Steuerung und Verwaltung von Sensoren und Manipulatoren als auch die korrekte Inbetriebnahme weiterer metrologischer Komponenten, wie Sensorwechseleinrichtung oder zusätzliche Rotationsachsen, obliegen gegenwärtig dem Anwender und wird von der zentralen Firmware der NPM-Maschine nicht berücksichtigt. Die daraus resultierende Dezentralisierung erfordert vom Nutzer viel Umsicht bei der Steuerung der Teilsysteme und umfassendes Detailwissen über deren Funktion. Das hat zur Folge, dass Softwarelösungen zur Steuerung von NPM-Maschinen und etwaigem Zubehör aus Effizienzgründen oftmals intransparent, applikationsspezifisch und schwer wiederverwendbar implementiert wurden. Damit einher geht unweigerlich eine hohe Wahrscheinlichkeit für schwerwiegende Programmier- und Bedienfehler, welche zur Beschädigung von Messobjekt und Messystem führen können.

Insbesondere die Umsetzung eines Multisensorkonzepts birgt ein beachtliches Potential, um die Flexibilität und Einsatzmöglichkeiten von nanometrologischen Messystemen, wie zum Beispiel NPM-Maschinen zu steigern und in Form von kombinierten Analysen mit unterschiedlichen Sensoren effizient und mit bisher nicht erreichter Präzision zum Tragen zu bringen. Durch den kombinierten Multisensor- bzw. Nanotool-Ansatz ist es im Rahmen einer einzigen Messung möglich, metrologische, physikalische sowie chemische Eigenschaften eines Messobjekts von der Makro- bis zur Nanoebene zu untersuchen und mit geeigneten Manipulatoren gezielt lokal zu verändern. Dafür ist es jedoch unerlässlich, ein neues, ganzheitliches und in sich geschlossenes Steuerungskonzept zu erarbeiten, um die universelle und transparente Nutzung von nanometrologischen Positionier- und Messsystemen und weiterer metrologischer Komponenten zu ermöglichen. Mit der vorliegenden Arbeit wird das Ziel verfolgt, die bislang voneinander losgelösten Komponenten in einem holistischen Modell zusammenzufassen und damit eine zentrale Steuerung der Module zu schaffen. Dieses Konzept soll das Positioniersystem, beliebige Sensorsysteme und metrologisches Zubehör, wie Sensorwechseleinrichtung oder zusätzliche Rotationsachsen zu einem Gesamtsystem im Sinne der iMERA zusammenfassen (siehe Abbildung 4).

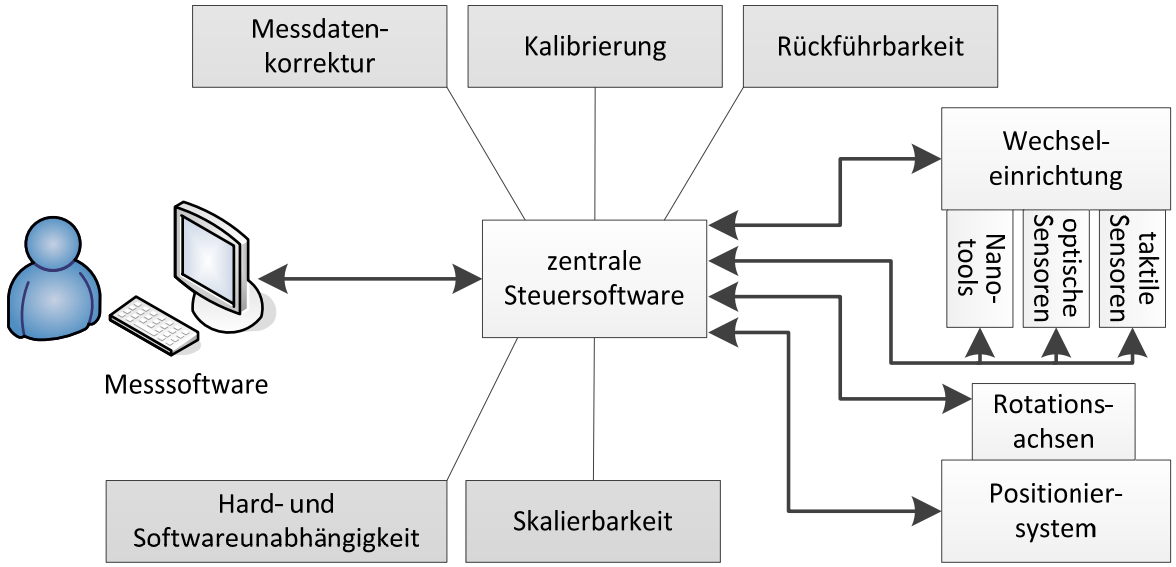


Abbildung 4: Anforderungen für das zu entwerfende Steuerungskonzept für nanometrologische Messysteme

Neben dem Zusammenführen dieser Systeme sind die komplexen Kalibrier- und Korrekturverfahren für die verschiedenen Sensorsysteme aus dem Verantwortungsbereich des Anwenders herauszulösen und konsistent in das Steuerungskonzept zu integrieren. Weitere Aspekte des zu entwickelnden zentralisierten Steuerungsmodells sind die einfache komponentenbasierte Skalierbarkeit des Gesamtsystems und die Fragestellung, inwieweit etablierte Schnittstellen zur Steuerung artverwandter makroskopischer Systeme adaptiert werden können.

Der Aufbau der Arbeit wurde dazu wie folgt gestaltet. Nach der Einführung des relevanten technischen Umfeldes erfolgt die Erarbeitung des umfassenden Steuerungsmodells für nanometrologische Positionier- und Messsysteme, basierend auf den technologischen Vorgaben der iMERA [9]. Das Konzept wird im weiteren Verlauf der Arbeit sukzessive zu einem Demonstrator weiterentwickelt und die daraus resultierende Steuerungssoftware für Nanopositionier- und Messmaschinen mit auf ihre Eignung für die Anwendung in der Nanometrologie hin untersucht. Um die hard- und softwareunabhängige Steuerung unter gleichzeitiger Bewahrung des universellen Funktionsumfangs von nanometrologischen Mess- und Positioniersystemen zu demonstrieren, werden verschiedene kommerzielle Steuerungsprogramme für makroskopische Koordinatenmessgeräte mit dem Demonstrator erprobt. Abschließend werden in Kapitel 7 die im Verlauf der Untersuchung gewonnenen Erkenntnisse zusammengefasst und diskutiert. Für die Planung und Modellierung der Software wird auf die Unified Modeling Language zurückgegriffen, welche auch in mehreren Abbildungen der Arbeit zur Anwendung kommt. Um die Interpretation dieser Grafiken zu erleichtern, ist in Anhang A für jeden verwendeten Notationstyp ein Beispiel hinterlegt. Darüber hinaus werden zentrale Termini der Softwareentwicklung in prägnanter Form im Glossar erläutert.

Kapitel 2 Stand der Technik

2.1 Nanomesstechnik

Die Grundidee der Nanomesstechnik geht auf einen von Feynman gehaltenen Vortrag von 1959 [11] zurück, worin er erstmals in Betracht zog, Moleküle und einzelne Atome zu manipulieren und so Materialien oder Maschinen aus atomaren Bestandteilen aufzubauen. Seit dem ausgehenden 20. Jahrhundert ist der technologische Stand hinreichend fortgeschritten, um Ansätze für Feynmans Vision erstmals zu realisieren. Die reproduzierbare Herstellung von Kohlenstoff-Nano-Röhrchen und kleinste mikro-mechanische Elemente sind nur der Anfang dieser Entwicklung und bis zur Herstellung autonomer Nanoroboter [12] sind noch viele Herausforderungen zu bewältigen. Aber auch für die gegenwärtigen Produkte der Nanotechnologie sind Methoden zur Qualitätssicherung erforderlich, da die meisten im Makrokosmos verwendeten Kenngrößen und Verfahren nicht uneingeschränkt auf die Nanometerskala übertragbar sind.

Das Schließen dieser methodischen Lücke obliegt der Nanomesstechnik als noch junger wissenschaftlicher Disziplin. Die Tendenz zur stetigen Verkleinerung makroskopischer Bauteile und die Notwendigkeit insbesondere der Halbleitertechnologie die Grundfläche mikroskopischer Strukturen zu vergrößern, führte historisch zu einer Annäherung von Koordinatenmesstechnik und Rasterkraftmikroskopie und legte damit den Grundstein für die Nanomesstechnik (siehe Abbildung 5). Zur Lösung metrologischer Aufgaben fließen neben den methodischen und technologischen Grundlagen dieser Disziplinen zusätzlich Ansätze der klassischen Mikroskopie ein, welche durch ein

wissenschaftliches Fundament neuartiger Konzepte für eine messtechnische Basis auf dieser Größenskala ergänzt werden. Insbesondere im Hinblick auf die messtechnischen, sensorischen Applikationen profitiert die Nanomesstechnik vom umfangreichen methodischen Umfang der drei Grundlagendisziplinen.

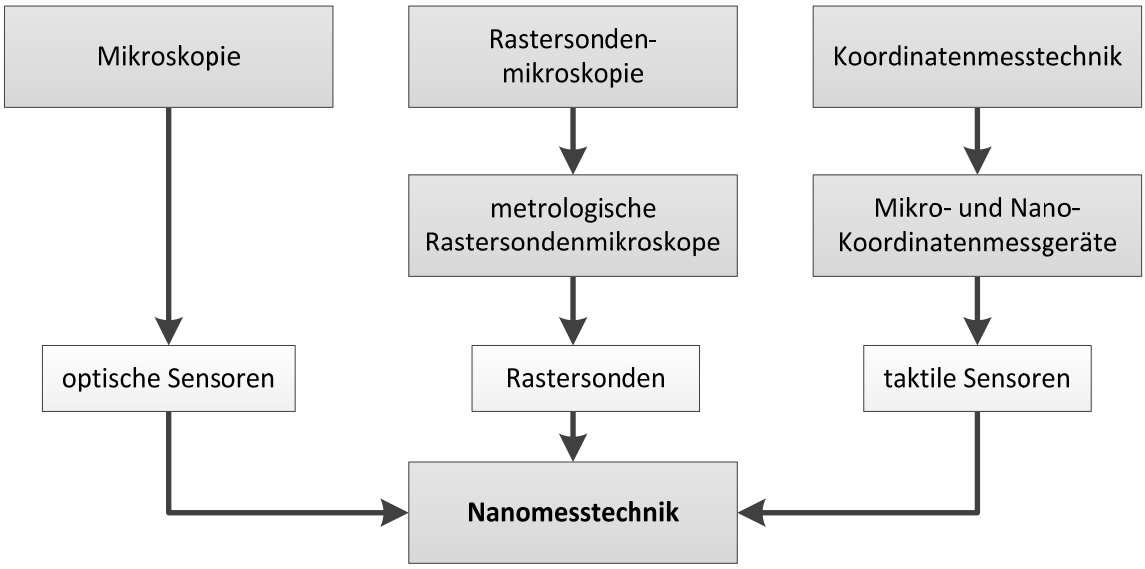


Abbildung 5: wissenschaftliche, technologische Einordnung der Nanomesstechnik

Für die messtechnische Erschließung der Mikro- und Nanometerskala entstanden bereits in den frühen 1990er Jahren erste Mikro- bzw. Nanokoordinatenmessgeräte und metrologische Rastersondenmikroskope (engl. atomic force microscope - kurz AFM). Obwohl Mikro- bzw. Nanokoordinatenmessgeräte bzw. metrologische AFMs mit Fokus auf verschiedene Anwendungen entwickelt wurden, weisen beide Messgeräteklassen im Hinblick auf Messbereich und erzielbarer Auflösung signifikante Ähnlichkeiten auf. Klassische AFMs ermöglichen mit bis zu 0,1 nm Auflösung üblicherweise Untersuchungen in einem lateralen Bereich von unter 100 x 100 µm². Diese Geräte sind in der dritten Dimension meist auf einen Messbereich von weniger als 10 µm beschränkt, ermöglichen aber eine subnanometeraufgelöste Analyse flächenhafter Objekte wie Halbleiterstrukturen oder mikromechanische Bauelementen. Im Gegenzug führte die Adaption klassischer Koordinatenmessgeräte (kurz KMG) für Objekte mit Ausdehnungen im (Sub-)Millimeterbereich zur Entstehung von Mikro- und Nano-KMGs bzw. NPM-Maschinen. Mit den Methoden der modernen Präzisionsfertigung hergestellte Objekte vereinen oft eine Überlagerung von Mikro- und Makrostrukturen, besitzen in der Regel jedoch eine geringere räumliche Ausdehnung. Somit sind bei Mikro- und Nano-KMGs dreidimensionale Stellbereiche zwischen mehreren 10 bis mehreren 100 mm je Achse hinreichend, um ein Messobjekt vollständig zu erfassen. Der maßgebliche Unterschied dieser Geräteklasse, insbesondere von NPM-Maschinen, zu den klassischen KMGs liegt in einem speziellen, fehlerminimierenden Aufbau, welcher es diesen Geräten ermöglicht, geringere Messunsicherheiten (zwischen 50 nm bis 3 nm) als ihre makroskopischen Vertreter zu erreichen.

Bereits an dieser Stelle tritt das heterogene technologische Umfeld der Nanomesstechnik zu Tage. Eine Vielfalt sensorischer Verfahren aus den drei Disziplinen der Mikroskopie, der Rastersondenmikroskopie und der Koordinatenmesstechnik stehen heterogenen Positionier- und Messsystemen mit divergenten konstruktiven und messtechnischen Ansätzen gegenüber. Die umfassende Kombinierbarkeit dieser Systeme eröffnet neue Möglichkeiten, den methodischen Umfang der Nanometrologie maßgeblich zu erweitern. Um das hier umrissene Bild der Nanomesstechnik weiter zu vertiefen und darauf aufbauend ein ganzheitliches Konzept für die Strukturierung einer neuartigen multisensoriellen NPM-Maschinen zu entwickeln, werden die

relevanten Konzepte der wissenschaftlichen Wurzeln der Nanometrologie im Folgenden detailliert erörtert.

2.2 Mikroskopie

Die von Ernst Abbe gelegten wissenschaftlichen Ursprünge der Mikroskopie gehen auf das Jahre 1873 zurück. Abbe gelang nach umfangreichen Experimenten die Entschlüsselung der Zusammenhänge bei der Bildentstehung am Mikroskop [13]. Nach seiner Theorie beeinflusst die Lichtbeugung am Objektiv wesentlich die Bildentstehung im Mikroskop. Der von Objektivöffnung und Arbeitsabstand festgelegte Halbwinkel (die Apertur) bestimmt, um welchen Winkel Licht gebeugt werden kann, um noch vom Objektiv erfasst zu werden. Die stets endliche Apertur führt dazu, dass ein idealer Punkt als Beugungsscheibchen abgebildet wird und somit das Auflösungsvermögen des optischen Systems begrenzt ist. Entsprechend dem Rayleigh-Kriterium lassen sich zwei Punkte sicher separieren, wenn die Zentren ihrer Beugungsscheibchen mindestens um den Radius r voneinander getrennt liegen [14]. Der Radius r ist dabei proportional zu Blendenzahl (Verhältnis aus Brennweite f zu Apertur D) und Lichtwellenlänge λ:

$$r = 1{,}2196 \cdot \frac{\lambda \cdot f}{D} \tag{2.1}$$

Im Bereich der optischen Mikroskopie sind ohne Spezialoptiken mit Licht im sichtbaren Spektrum nur laterale Auflösungen bis etwa 500 nm erzielbar. Die vertikale bzw. axiale Auflösung optischer Sensoren ist nicht durch die Beugung begrenzt, jedoch ist das Auflösungsvermögen abhängig vom Messprinzip um den Faktor 5 bis 10 höher. Die hohe Auflösung entlang der Vertikalen macht optische Messverfahren insbesondere für die Anwendung in der Nanomesstechnik interessant. Moderne Immersionsoptiken arbeiten statt in Luft mit Medien höherer Brechzahl, wodurch nummerische Aperturen größer als $NA = 1{,}2$ erreicht werden. Durch ihre Anwendung ist die laterale Auflösungsgrenze in den entsprechenden Immersionsmedien deutlich kleiner als die oben genannten 500 nm.

Seit mehreren Dekaden wird der Mensch als Detektor und Beurteilungssystem in der Mikroskopie zugunsten einer besseren Reproduzierbarkeit der Ergebnisse zurückgedrängt und es kommen vermehrt technische Messsysteme bei Untersuchungen zum Einsatz. In diesem Zusammenhang entstanden verschiedene optische Messverfahren und Sensorsysteme, welche auch Auflösungen im Nanometerbereich ermöglichen. Diese Fähigkeit und das berührungslose Messen mit Licht führt zu der großen Bedeutung der optischen Sensoren für die Nanomesstechnik.

Optische Sensoren

Die Funktionsweise optischer Sensoren allgemeingültig zu beschreiben ist nur bedingt möglich, da der Aufbau optischer Sensoren in Abhängigkeit vom realisierten Messprinzip variiert. Die prinzipielle Struktur eines optischen Sensors besteht jedoch stets aus einer Lichtquelle und Abbildungsoptiken, welche das emittierte Licht auf die Oberfläche des Messobjekts lenkt bzw. das reflektierte Licht bündelt und auf den bzw. die photoelektrischen Detektoren lenken. Die optische Antastung erfolgt in der Regel durch eine Scanbewegung entlang der optischen Achse des Sensors. Die Berechnung der Raumkoordinaten der angetasteten Oberfläche erfolgt unter Einbeziehung der Sensorgeometrie, dem Ansprechen der photoelektrischen Detektoren und je nach Messprinzip von weiteren Parametern. Einige Prinzipien, wie das Verfahren der Laserfokusvariation [15] nutzen kohärente Lichtquellen, andere breitbandiges inkohärentes Licht (z. B. Weißlichtinterferometrie). Da optische Messverfahren keinen physischen Kontakt zum Messobjekt herstellen, ist es möglich, eine Vielzahl von

Punktsensoren zu Zeilen- oder Flächensensoren zusammenzufassen, ohne dass sich die Einzelsensoren gegenseitig beeinflussen. Mit dem so entstehenden Arraysensor können, im Vergleich zu den taktilen Punktsensoren, mit einer Messung große Oberflächenbereiche hoch aufgelöst erfasst werden (siehe Abbildung 6). Ein Vorteil optischer Sensoren ist das berührungslose Messen [16]. Dadurch können Gestaltuntersuchungen auch an weichen oder empfindlichen Objekten durchgeführt werden [17], welche aufgrund der Antastkräfte taktiler Verfahren verformt oder beschädigt würden. Weiterhin ermöglichen optische Sensoren neben der parallelen Datenerfassung auch die Untersuchung sehr kleiner Merkmale.

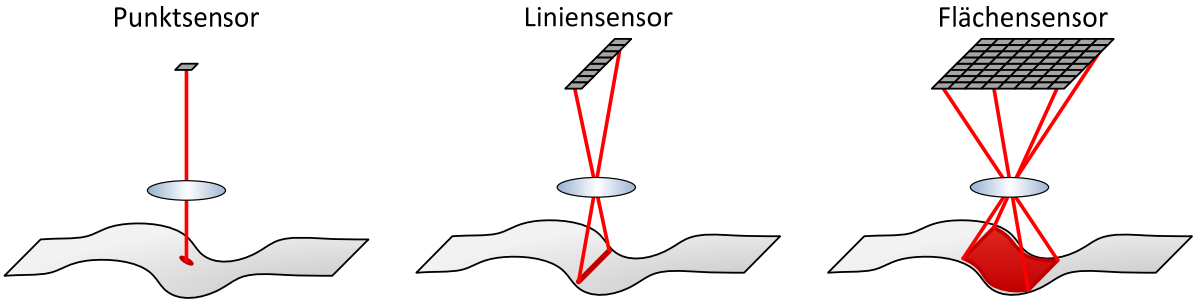


Abbildung 6: Prinzipieller Aufbau optischer Sensoren

Optische Antastverfahren unterliegen aber auch Einschränkungen. Senkrechte Kanten oder Hinterschneidungen können nicht angetastet werden. An Flanken mit steilen Anstiegen wird nur eine unzureichende Lichtmenge zum Photodetektor reflektiert, so dass aus dem empfangenen Signal keine verlässlichen Informationen zur Position der Oberfläche ermittelt werden können. Neben der äußerlichen Formgebung des Objektes spielen auch Materialeigenschaften eine bedeutende Rolle. Eigenschaften wie Farbgebung, Texturierung oder Streuverhalten, welche bei einer taktilen Antastung nicht relevant sind, haben einen deutlichen Einfluss auf das Messergebnis und die resultierende Abweichung ist in der Regel nur schwer abschätzbar. Insbesondere sind spiegelnde, volumenstreuende oder stark texturierte Oberflächen mit optischen Systemen zum Teil nur unter Schwierigkeiten und Inkaufnahme großer Unsicherheiten zu vermessen [16]. Die Hauptschwierigkeit bei der Nutzung optischer Sensoren liegt für den Anwender darin, ein Sensorsystem auszuwählen, dessen Messprinzip für die Erfassung der Messobjektoberfläche geeignet ist. So muss unter Umständen für die gleiche Messaufgabe bei Objekten mit unterschiedlichen Materialeigenschaften ein anderes Sensorsystem gewählt werden. Ferner sind optische Sensoren nur in der Lage, die Antastung entlang der optischen Achse auszuführen. Trotz dieser Einschränkung sind optische Sensoren für die Nanomesstechnik unverzichtbar, da diese Sensoren als parallelmessender Flächensensor ausgeführt werden, mit einer Messung großflächige Bereiche stark strukturierter Oberflächen berührungslos erfassen können. Insbesondere optische Messverfahren mit einem sehr hohen Auflösungsvermögen kommen in der Nanomesstechnik zur Anwendung.

Grundlegend wird bei den optischen Messverfahren zwischen der lateralen und der axialen Auflösung unterschieden. Die laterale Auflösung ist durch die Apertur des optischen Systems auf ca. 500 nm begrenzt, wohingegen die koaxiale Auflösung deutlich höher sein kann. In Tabelle 1 sind optische Messverfahren aufgeführt, welche die Erfassung von 2,5D-Daten[1] mit einer hinreichend hohen Auflösung für nanometrologische Applikationen ermöglichen.

[1] Im Rahmen dieser Arbeit wird unter 2,5D-Daten eine Datenrepräsentation verstanden, bei welcher der Basis eines zweidimensionalen (äquidistanten) Rasters jedem Rasterpunkt ein eindeutiger Wert für die dritte Dimension zugeordnet ist.

optische Messverfahren	axiale Auflösung [nm]	Sensortyp
Depth From Focus **[18]**	50	Flächensensor
Weißlichtinterfererenzmikroskopie **[19]**	0,1-1	Punkt-/Linien-/Flächensensor
Konfokalmikroskopie **[20]**	1-10	Punktsensor
Laserfokussensor **[21]**	0,1-1	Punktsensor

Tabelle 1: Für die Nanomesstechnik geeignete optische Messverfahren

Die Weißlichtinterferenzmikroskopie (kurz WLI), als Beispiel für ein optisches Messverfahren, ist für nanometrologische Anwendungen besonders geeignet. Das Verfahren kann als hoch paralleler Flächensensor implementiert werden und ermöglicht dem Anwender mit einer geringen Messzeit berührungslos große Bereiche eines Prüflings zu erfassen, wodurch ausgedehnte und empfindliche Objekte ohne die Gefahr ihrer Beschädigung großflächig analysiert werden können. Die Integration optischer Sensoren in nanometrologische Messgeräte erfordert allerdings die Berücksichtigung der Spezifika der optischen Messverfahren. Insbesondere Messmethoden, welche bei Linien- und Flächensensoren zur Anwendung kommen, erfordern spezielle Positionierregime für die Datenerfassung. Diese müssen vom Anwender oder der Steuerungssoftware bereitgestellt werden. Im Rahmen der Arbeit wird in Kapitel 6 ein Weißlichtinterferenzmikroskopsensor für nanometrologische Messungen eingesetzt und in diesem Zusammenhang auf dessen Besonderheiten detailliert eingegangen.

2.3 Rastersondenmikroskopie

Im Jahr 1981 wurde das Rastertunnelmikroskop (engl. Scanning Tunneling Microscope, kurz STM) erstmals von Binnig [22] vorgestellt. STMs waren die erste Ausprägung von Rastersondenmikroskopen (engl. Scanning Probe Microscope, kurz SPM). SPMs sind funktionell vergleichbar mit makroskopischen Oberflächenprofilometern, jedoch sind sie in der Lage, atomare Strukturen aufzulösen. Maßgeblich für das Messverfahren eines Rastertunnelmikroskops ist, dass ein Cantilever mit einer sehr feinen Siliziumtastspitze (Apexradien von 10 nm bis 100 nm) von einem 3D-Positioniersystem über die Oberfläche des Messobjekts geführt wird. Dabei werden die Antastpunkte auf der Objektoberfläche lateral äquidistant abgerastert und dabei die z-Position der Objekttopographie bestimmt. Während der Rastermessung wird eine Regelgröße, welche mit der Entfernung zwischen Sonde und Objekt korreliert ist, konstant gehalten. Im Falle des Binning-STM ist diese Regelgröße der Tunnelstrom zwischen Sonde und Messobjekt. Das Ausgleichen etwaiger Regelabweichungen während der Messung erfolgt durch das kontinuierliche Nachführen des Objekts entlang der z-Richtung. Damit korreliert der Ausgang des Positionsreglers mit der Topographie des Messobjekts und die z-Position kann durch die Modellierung der physikalischen Zusammenhänge bestimmt werden. Da SPMs einen Tunnelstrom zur Bestimmung der Distanz zwischen Sonde und Messobjekt verwenden, müssen die Messobjekte elektrisch leitfähig sein, was nicht bei allen Objekten vorausgesetzt werden kann. Eine weitere SPM-Unterklasse, die Rasterkraftmikroskope (engl. Atomic Force Microscope, kurz AFM) nutzen statt des Tunnelstroms die Durchbiegung des Sondencantilevers als Regelgröße und ermöglichen damit Messungen an leitenden und nichtleitenden Oberflächen. Ferner ist es mit speziellen AFM-Sonden oder -Messverfahren möglich, gezielt atomare Kräfte mit unterschiedlicher Ausprägung und Reichweite (siehe Tabelle 2) isoliert zu betrachten. Dadurch wird neben der Messung topografischer Eigenschaften eines Objekts auch die Untersuchung physikalischer, elektrischer oder chemischer Eigenschaften ermöglicht.

Wechselwirkung	Reichweite [nm]
Pauli-Abstoßung	0,1
Coulomb-Abstoßung	0,1
Kovalente Bindung	0,1
Metallische Bindung	< 1
Ionenbindung	< 1
Kapillarkräfte	< 10
Van der Waals Kräfte	< 100
Elektrostatische Kräfte	> 100
Magnetische Kräfte	> 100

Tabelle 2: Reichweite atomarer Wechselwirkungskräfte **[23]**

Der prinzipielle Aufbau eines AFMs ist in Abbildung 7 dargestellt. Auf die verspiegelte Rückseite des Cantilevers wird ein Laserstrahl fokussiert, welcher von dort auf eine Photodiode reflektiert wird. Der Strahl wird vor der Messung mittig auf dem Photodetektor justiert, so dass der Photodifferenzstrom bei vorgegebener Kraft gegen Null geht. Während der Messung wird die Probe unter der Sonde verfahren. Die Oberflächenstruktur bewirkt ein Verbiegen des Cantilevers, wodurch der Laserstrahl abgelenkt wird, was einen messbaren Photodifferenzstrom erzeugt. Der Differenzstrom stellt zugleich die Regelgröße des Regelkreises Sonde-Photodetektor-z-Positionierung dar und wird vom Regler minimiert. Damit wird der Cantilever auf einer konstanten Durchbiegung gehalten und das Ausgangssignal des Reglers stellt eine mit der Topographie korrelierte Größe dar. In Abhängigkeit vom verwendeten Spitzenapex des Cantilevers, des Antastregimes (siehe Tabelle 3) und der als Regelgröße verwendeten atomaren Wechselwirkung können AFMs laterale Strukturen von wenigen Nanometern auflösen, was sie als Werkzeug für Untersuchungen an mikro- und nanostrukturierten Oberflächen qualifiziert. Klassische AFMs verwenden Piezoaktoren als 3D-Positioniersystem. Diese Röhrenpiezoscanner bewegen die Probe auf einem Kreissegment, woraus sich große Führungsfehler und ein Übersprechen der Achsen ergeben [24]. Darüber hinaus verschlechtern Nichtlinearitäten, Hysterese und Kriechen des Piezos die Reproduzierbarkeit weiter.

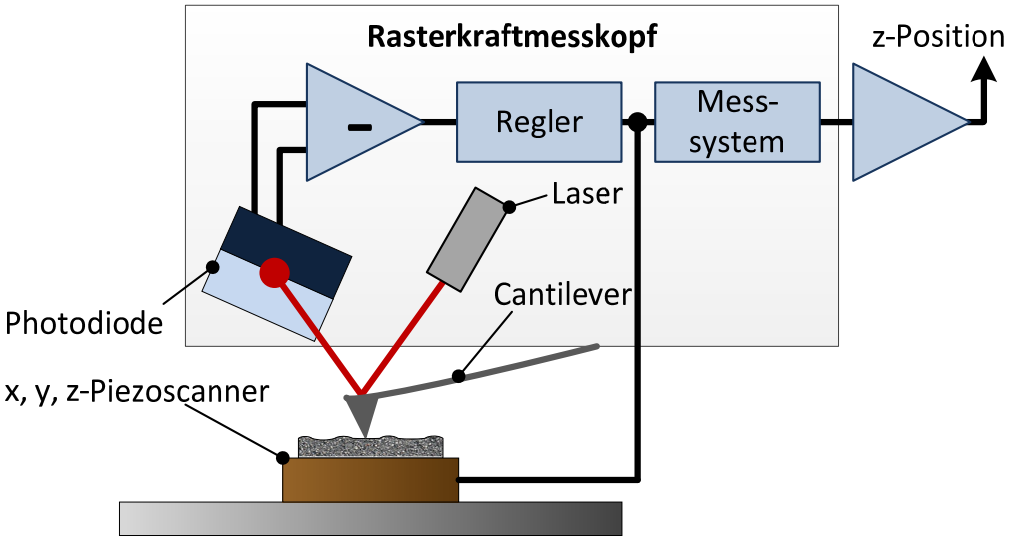


Abbildung 7: Aufbauschema eines Rasterkraftmikroskops

AFM-Modus	Beschreibung
Contact Mode	Die Sonde wird während des Scans in permanentem Kontakt mit der Objektoberfläche gehalten. Es dominieren die Nahbereichskräfte, wodurch hohe laterale Auflösungen erreicht werden. Durch die hohen Kontaktkräfte tritt verstärkt Abnutzung bzw. Kontamination der Sonde auf.
Non-Contact Mode	Die Sonde wird durch Rückkopplung zur Schwingung mit ihrer Eigenfrequenz angeregt und in einem Abstand von 10 nm bis 100 nm über der Oberfläche entlanggeführt. Änderungen der atomaren Wechselwirkungskräfte wirken sich auf die Resonanzfrequenz aus, welche als Eingangsgröße für die z-Regelung dient. Der Betriebsmodus zeichnet sich durch sehr geringe Abnutzung, aber nur mäßige laterale Auflösung aus.
Tapping Mode	Ist eine Mischform aus Contact und Non-Contact Mode. Die Sonde wird zur Schwingung nahe der Eigenfrequenz angeregt, berührt die Oberfläche jedoch nur im unteren Umkehrpunkt. Änderungen der atomaren Wechselwirkungen modulieren die Amplitude der Schwingung. Die Auflösung in diesem Betriebsmodus ist mit dem Contact Mode vergleichbar, wobei der Verschleiß geringer ausfällt.

Tabelle 3: Grundlegende AFM-Betriebsmodi

Metrologische Rasterkraftmikroskope

Metrologische Rasterkraftmikroskope sind eine Weiterentwicklung klassischer AFMs. Aufbau und Funktionsweise von metrologischen AFMs und klassischen SPMs sind abgesehen vom Positioniersystem jedoch weitgehend identisch. Wegen der schlechten messtechnischen Eigenschaften von Piezoscannern wird für den Aufbau metrologischer AFMs eine komplexere lineare, achsparallele Positioniereinrichtung mit hoher Führungsgüte und Verstellwegen von zum Teil mehreren Millimetern verwendet (siehe Abbildung 8), welche von speziellen hochgenauen Messsystemen überwacht wird. Metrologische AFMs werden für Messungen und Analysen planarer strukturierter Oberflächen eingesetzt, weshalb der Verstellbereich der z-Achse wie bei klassischen AFMs nur wenige µm umfasst.

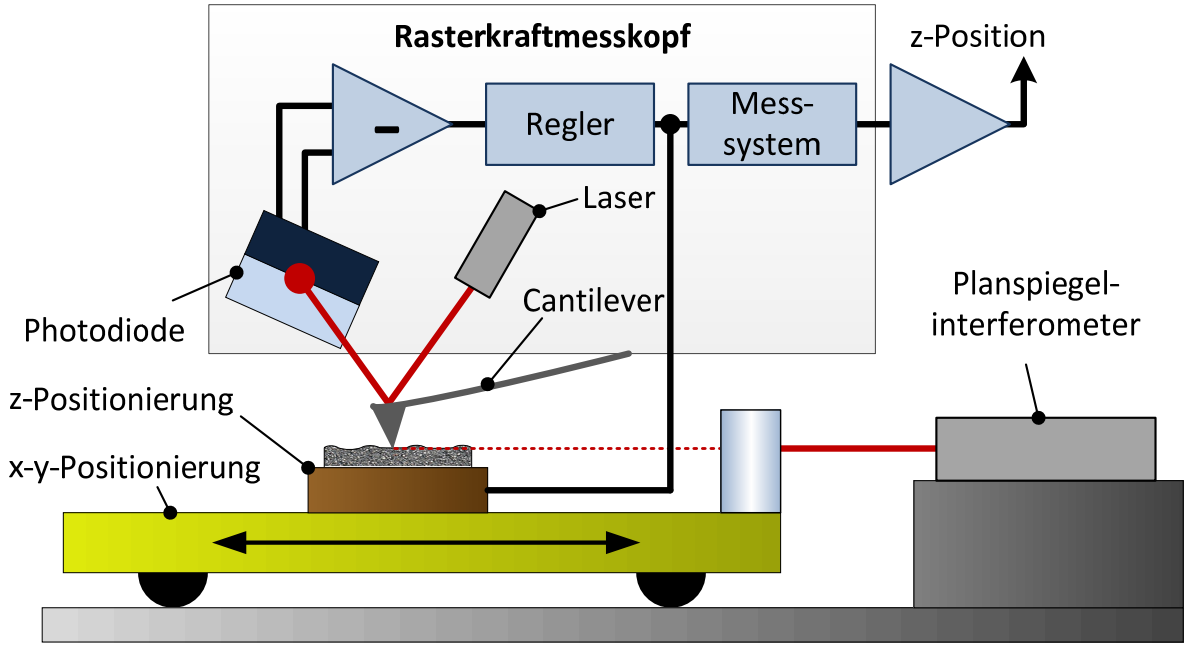


Abbildung 8: Funktionsschema eines metrologischen Rasterkraftmikroskops (nach [24])

Die Überwachung der aktuellen Position wird durch Längenmesssysteme mit geeigneten Längenmaßstäben bewirkt. Hierfür werden oft Interferometeraufbauten oder Glasmaßstäbe verwendet, wie sie verschiedene Institutionen in ihren Aufbau metrologischer AFMs verwenden (siehe Tabelle 4). Sowohl die Sensoren als auch die Positioniersysteme metrologischer AFMs stellen einen bedeutenden Teilaspekt dieser Arbeit dar. Da sie wichtige Werkzeuge der Nanomesstechnik sind, sollen diese Systeme beim Entwurf des holistischen Steuerungskonzepts in Kapitel 3 besondere Berücksichtigung finden.

metrologische AFMs	Einrichtung
Sub-Atomic Measuring Machine **[25]**	University of Carolina, Massachusetts Institute of Technology
Molecular Measuring Machine **[26]**	National Institute of Standards and Technology
NanoSurf IV **[27]**	National Physical Laboratory
Metrologie-Rasterkraftmikroskop **[24]**	Eidgenössisches Institut für Metrologie

Tabelle 4: Aufzählung metrologischer Rastersondenmikroskope

2.4 Koordinatenmesstechnik

In makroskopischen Skalengrößen stellt die Koordinatenmesstechnik die Grundlagen für die Maßhaltigkeitsbewertung im Rahmen von wissenschaftlichen Untersuchungen und für die Qualitätssicherung dar. So findet die Koordinatenmesstechnik Anwendung in Forschung & Entwicklung, in Metrologieinstituten [28] sowie in der Industrie, sei es losgelöst von der Fertigung als Eingangskontrolle oder prozessbegleitend in speziellen Messräumen oder als Inline-Messung [29] im laufenden Fertigungsprozess. Die Koordinatenmesstechnik als Teildisziplin der Fertigungsmesstechnik zeichnet sich dadurch aus, dass Eigenschaften des Messobjekts mit Punktmessungen erfasst werden und der quantitative Wert rechnerisch durch diese Daten bestimmt und nicht direkt am Objekt abgelesen wird. Dieser Zusammenhang wird am Beispiel in Abbildung 9

verdeutlicht. Die Vorgabe für die Fertigung ist die vom Konstrukteur festgelegte Gestalt eines Objekts (a), anhand welcher das Objekt gefertigt wird. Da der Fertigungsprozess permanent Schwankungen und Störeinflüssen (z. B. Verschleiß von Werkzeugen) unterliegt, erfüllt das gefertigte Teil die konstruktive Vorgabe nur mit einer gewissen Toleranz (b). Um die Maßhaltigkeit des Werkstücks zu beurteilen, wird dieses an verschiedenen Stellen mit einem Koordinatenmessgerät (kurz KMG) angetastet (c) und aus den gemessenen Punkten für jedes Geometrieelement der konstruktiven Vorgabe ein zugehöriges Ausgleichselement [30] mit numerischer Toleranz bestimmt (d). Die Beurteilung der Maßhaltigkeit erfolgt schließlich auf Grundlage der Toleranzen der Ausgleichselemente und ihrer räumlichen Beziehung zueinander (e).

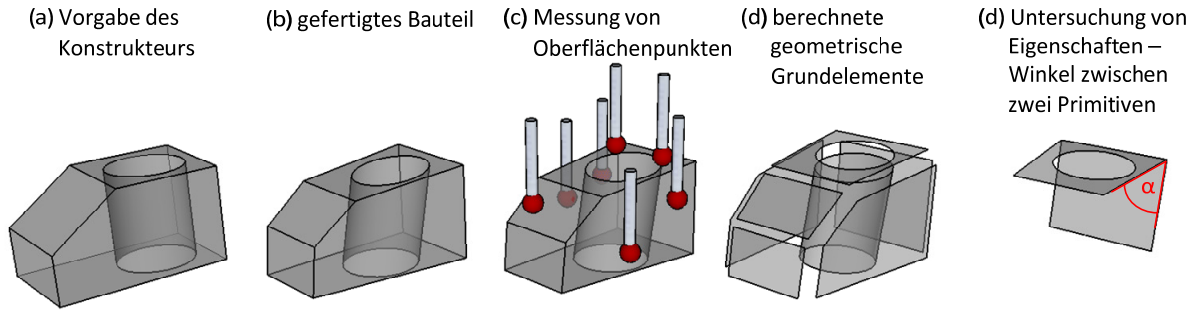


Abbildung 9: Anwendung der Koordinatenmesstechnik an einem Beispiel (in Anlehnung an [28] & [31])

Vielfach bilden Computer-Aided-Design-Daten (CAD) die konstruktiven Vorgaben in der modernen Fertigung. Die damit beschriebene Formgebung basiert in der Regel ebenfalls auf tolerierten geometrischen Grundelementen. Ein Vergleich dieser Vorgabe mit den am Objekt ermittelten Formelementen ermöglicht die einfache Beurteilung der Maßhaltigkeit sowie von Form- und Lageabweichungen. Die Methode der Koordinatenmesstechnik, Oberflächenpunkte am Objekt zu messen und die Qualitätsbeurteilung auf einer abstrakten Ebene vorzunehmen, macht das Verfahren unabhängig von der Formgebung des Messobjekts. Dadurch sind Koordinatenmessgeräte für metrologische Anwendungen universell einsetzbar und ein unverzichtbares Werkzeug der Fertigungsmesstechnik [32].

KMGs bestehen prinzipiell aus einem Tastsystem und mindestens drei unabhängig voneinander beweglichen Achsen mit Längenmaßstäben, welche durch das Chassis fixiert werden (siehe Abbildung 10). Die Längenmaßstäbe sind meist räumlich orthogonal angeordnet und bilden das intrinsische Koordinatensystem des KMGs. Durch die Superposition von Bewegungen entlang der Achsen kann das Tastsystem des KMGs frei im gesamten Messvolumen positioniert werden. Die Erfassung eines Objektpunktes am Prüfling erfolgt durch das Herbeiführen einer Berührung von Messobjekt und Tastsystem. Das Tastsystem verfügt über ein Detektorsystem für jede Raumachse, welche die Antastkraft bei der Berührung registrieren und an eine Steuerungshardware weiterleiten, welche daraufhin die Speicherung der aktuellen Achspositionen als Messpunkt veranlasst. Die Klassifizierung (taktiler) Sensoren erfolgt entsprechend des Detektoraufbaus als Schalter oder Messsystem. Entsprechend ist das Ergebnis der Wandlung ein qualitatives bzw. ein quantitatives Signal. Aufgrund der qualitativen Aussage über die Antastkraft erzielen messende Sensoren deutlich geringere Messunsicherheiten und ermöglichen zudem die Datenerfassung im Scanmodus. Bei einer Scanmessung wird das Tastsystem mit dem Prüfling in Kontakt gebracht und mit konstanter Kraft über die Oberfläche geführt. Während der Bewegung werden fortlaufend Messpunkte aufgezeichnet, wodurch der Scanmodus im Vergleich zur Einzelpunktantastung die Messzeit drastisch reduzieren kann. Neben dem oben dargelegten Grundaufbau verfügen moderne KMGs über vielfältige Erweiterungen, wie Tasterwechseleinrichtungen, zusätzliche Rotationsachsen, Sensoren mit Dreh-Schwenk-Einrichtungen und ermöglichen eine schnelle Datenerfassung durch Scanmessungen.

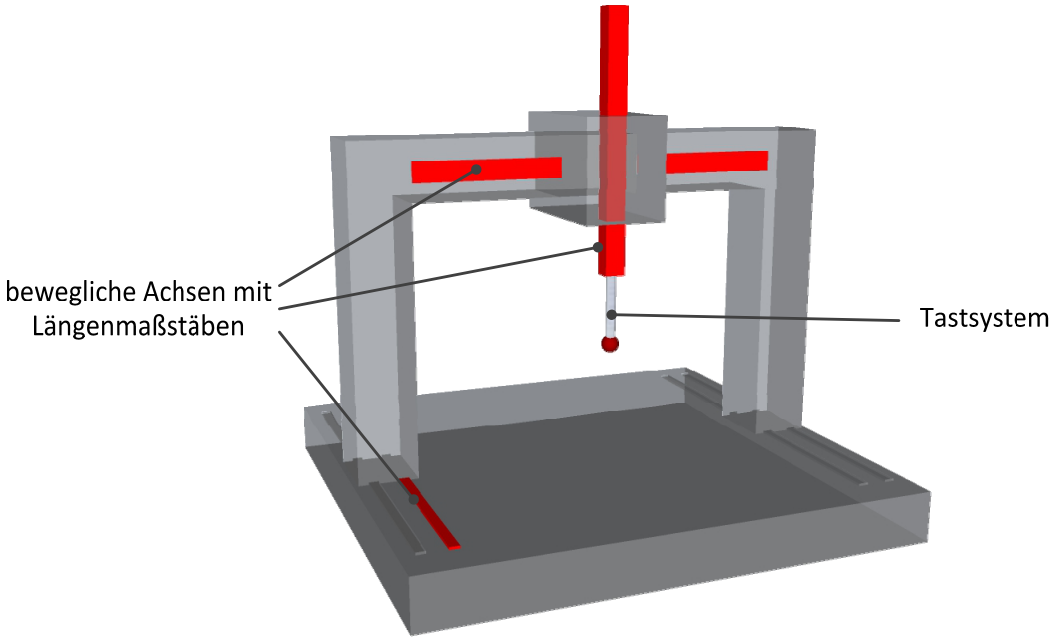


Abbildung 10: Prinzipieller Aufbau eines KMGs

Die Koordinatenmesstechnik ist ein Grundpfeiler der Nanomesstechnik und legt damit auch eine Basis für metrologische Untersuchungen auf der Mikro- und Nanometerskala. Viele Verfahrensweisen und Systemlösungen und bilden eine Grundlage für Methoden und Ansätze der Nanomesstechnik. Jedoch können die makroskopischen Ansätze nicht ohne weiteres in den Nanokosmos übertragen werden. Der Aufbau von KMGs, Sensorsystemen als auch Mess- und Korrekturverfahren führen zu Messunsicherheiten, welche im Makroskopischen vernachlässigt werden können, aber aus Sicht der Nanomesstechnik nicht vertretbar sind. Für die Lösung dieser Probleme mussten in der Nanometrologie neue Ansätze gefunden werden. Die betreffenden Systeme (Sensorwechseleinrichtung und Kipp-/Drehtisch für NPM-Maschinen) und ihre nanometrologischen Besonderheiten werden ab dem Abschnitt 2.4.2 detailliert erläutert.

Ebenso wie in der KMT ist geeignete Software für die Steuerung der Messhardware auch für die Nanometrologie von besonderer Bedeutung. Die Messdatenverarbeitungs- und Steuerungssoftware ermöglicht sowohl die Messung als auch die Auswertung der gewonnenen Daten zu automatisieren und gleichzeitig den Bediener als Fehlerquelle zu eliminieren. Von zentraler Bedeutung für die Steuerbarkeit der Hardware ist jedoch die Festlegung einer verbindlichen Steuerungsschnittstelle. Inspection plus plus for Dimensional Measurement Equipment (kurz: I++DME) stellt im Umfeld der KMT eine entsprechende, allgemein akzeptierte und herstellerunabhängige Steuerungsschnittstelle dar. Aufgrund der hohen Akzeptanz und Verbreitung von I++DME soll diese Schnittstelle im Rahmen dieser Arbeit für die Nanometrologie portiert werden. Dadurch kann die Steuerung von NPM-Maschinen ebenso wie in der makroskopischen Messtechnik vereinheitlicht werden. Zu diesem Zweck wird neben der folgenden Darstellung von I++DME die Realisierbarkeit der Portierung in Kapitel 4.2 tiefgreifend untersucht.

2.4.1 Die KMG-Schnittstelle Inspection++ for Dimensional Measurement Equipment

Ursprünglich existierte keine einheitliche Schnittstelle zur Steuerung von KMGs, weshalb jeder Hersteller seine eigene Steuerungssoftware entwickelte und diese mit seinen KMGs vertrieb. Auf Druck der Anwender einigten sich die KMG-Hersteller im Jahr 2000 auf eine einheitliche Schnittstelle mit grundlegenden Basisbefehlen zur Steuerung von KMGs [33]. In mehreren Folgerevisionen wurde die Unterstützung für zusätzliche metrologische Hardware wie Rotationsachsen, schwenkbare

Sensoren und Sensorwechseleinrichtungen aufgenommen, wodurch die Schnittstelle ihren universellen Charakter erhielt und ein hohes Maß an Interoperabilität gewährleistet. Die Harmonisierungsinitiative Inspection plus plus for Dimensional Measurement Equipment (I++DME) avancierte innerhalb weniger Jahre zu einem allgemein anerkannten Standard zur Steuerung von KMGs und wird von nahezu allen KMG-Herstellern (u. A. Nikon Metrology, Zeiss Industrielle Messtechnik und Hexagon Metrology) und Anbietern von Messsoftware (Messtechnik Wetzlar mit Quindos7, Wenzel mit Metromec, Carl Zeiss IMT mit Calypso oder Renishaw mit MODUS) unterstützt [34]. Die einheitliche Schnittstelle des OSIS-Proxy ermöglicht den einfachen Austausch von Soft- und Hardware, eine objektive Beurteilung der Leistungsfähigkeit von KMGs und Steuerungssoftware verschiedener Hersteller und versetzt den Anwender so in die Lage, die für eine Messaufgabe optimale Kombination von KMG-Hardware und Steuerungssoftware auszuwählen. Durch den Quasistandard I++DME [35] ist der Anwender nicht mehr an die proprietären Schnittstellen der Messgerätehersteller gebunden und kann eine einzige Steuer- und Messsoftware für unterschiedliche Koordinatenmessgeräte verwenden. Durch die Unabhängigkeit von Inspektionsgerät und Bediensoftware wird der Austausch von Messgeräten und Software vereinfacht.

Logische Platzierung der I++DME-Schnittstelle

Eine Schnittstelle wird im Rahmen der I++DME-Dokumente als Software festgelegt, welche alle Teilsysteme des physischen KMGs in einer Vielzahl von Parametern in einem definierten Modell verwaltet und die Steuerung der KMG-Hardware realisiert (siehe Abbildung 11). Von der logischen Platzierung der Schnittstellensoftware zwischen Anwender und KMG leitet sich auch die englische Bezeichnung Middleware ab. Die generelle Aufgabe der Middleware ist die transparente Vermittlung von Diensten zwischen verschiedenen Anwendungen [36]. Durch die Einführung der vermittelnden I++DME-Schnittstelle wird das hinter der Schnittstelle befindliche KMG zu einem generischen I++DME-Gerät abstrahiert, welches vom Nutzer über das Protokoll gesteuert wird. Der Anwender benötigt für die indirekte Steuerung des KMGs somit nur Wissen über die Schnittstelle. Diese übersetzt Nutzeraufrufe in für die KMG-Hardware verständliche Steuersequenzen, wodurch die eigentliche Steuerung der KMG-Hardware für den Nutzer unsichtbar bzw. transparent wird und es damit möglich ist, die KMG-Hardware ohne Auswirkungen auf die Nutzerschnittstelle auszutauschen.

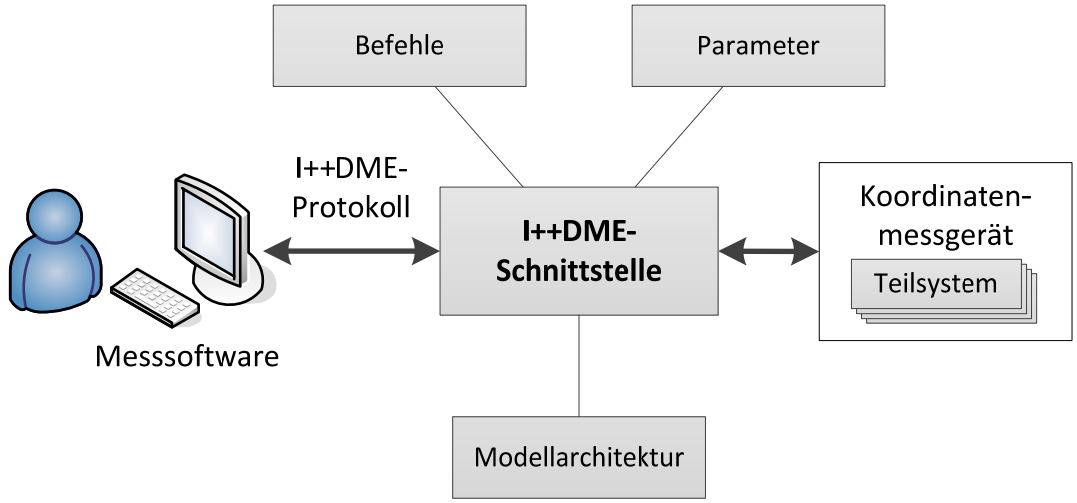


Abbildung 11: Interaktionsmöglichkeiten I++DME-Anwender und -Messgerät (nach [35])

Die Kommunikation zwischen Nutzer und I++DME-Schnittstelle basiert auf dem Client-Server-Prinzip (engl. client: Nutzer oder Anwender, server: dienstbereitstellendes System). Die I++DME-Schnittstelle stellt diverse Dienste in Form von Messbefehlen für den Anwender bereit. Syntax, Semantik und die zeitliche Abfolge des Datenaustauschs zwischen Anwender und Middleware werden durch das I++DME-Protokoll definiert, wodurch die Zuordnung von Befehlen und Daten im

asynchronen Kommunikationskanal ermöglicht wird. Die Grundlage für den Datenaustausch zwischen I++DME-Schnittstelle und Anwender bilden textbasierte Befehls- und Antwortsequenzen mit festen Parametersätzen, welche über das hardwareunabhängige verbindungsorientierte Transmission Control Protocol (TCP) übertragen werden. Dadurch kann ein etwaiger Verlust bei der Übertragung von Befehlen oder Messdaten ausgeschlossen werden.

2.4.2 Mikro- und Nano-Koordinatenmessgeräte

Mikro- und Nano-KMGs sind dimensionelle Messsysteme mit besonderen Anforderungen an die Präzision und die Messunsicherheit. In Abhängigkeit vom verwendeten Sensorsystem, ermöglichen sie punktbasierte Koordinatenmessungen, optische Flächenmessungen oder flächenhafte Messungen im Sinne der Rastersondenmikroskopie und sind damit in der Lage, Objektstrukturen bis auf die atomare Ebene aufzulösen. Die Auflösung des Sensorsystems allein ermöglicht jedoch noch keine Aussage über die Qualität der vom Gesamtsystem gemessenen Daten. Störungen und Unsicherheiten aller Teilsysteme des Mikro- bzw. Nano-KMGs tragen zu dessen Gesamtmessunsicherheit bei (siehe Tabelle 5), wodurch diese oft nur wenig geringer als 100 nm ist.

Mikro- bzw. Nano-KMG	Institut / Hersteller	Messunsicherheit [nm]
ISARA **[37]**	IBS/CFT Philips	30
ISARA 400 **[38]**	IBS	100
Nano Coordinate Measuring Machine **[39]**	National Taiwan University	< 30
Special Coordinate Measuring Machine **[40]**	Physikalisch Technische Bundesanstalt	< 100
F25 **[41]**	TU Eindhoven	250
TriNano Ultra Precision Coordinate Measuring Machine **[42]**	Xpress Precision Engineering	≥ 100
Small Coordinate Measuring Machine **[43]**	National Physical Laboratory	< 90
Nanopositionier- und Nanomessmaschine **[44]**	Technische Universität Ilmenau	3

Tabelle 5: Aufzählung von Mikro- und Nanokoordinatenmessgeräten

Insbesondere der Aufbau des mehrachsigen Positioniersystems erhält neben der Qualität der Führungen, der Präzision der Antriebe und der Genauigkeit der Längenmesssysteme einen bedeutende Rolle für die erreichbare Gesamtmessunsicherheit. Die einfache Skalierung makroskopischer KMG-Aufbauten ist für die Realisierung von Mikro- und Nano-KMGs ungeeignet, da aus dem Aufbau herrührende Unsicherheiten nicht mit dem Aufbau skalieren. Um der Anforderung nach einer geringen Messunsicherheit gerecht zu werden, muss der Aufbau des Gesamtsystems umgestaltet werden und dem Abbe'schen Komparatorprinzip genügen.

Die an der TU Ilmenau entwickelten Nanopositionier- und Nanomessmaschinen folgen dieser Maßgabe, was es ihnen in Verbindung mit ihrem hoch präzisen Positioniersystem ermöglicht, Messunsicherheiten im unteren Nanometerbereich zu erlangen. NPM-Maschinen bilden damit eine eigene Geräteklasse, wobei sie funktional jedoch mit Mikro- und Nano-KMGs vergleichbar sind.

2.4.2.1 Abbe-Komparatorprinzip

Abbe erkannte 1890 [45] den Einfluss der Anordnung von Messobjekt und Maßverkörperung auf die Reproduzierbarkeit eines Messsystems. Ihm folgend muss

> „die Anordnung [von Messobjekt und Maßstab] stets so erfolgen, dass die zu messende Strecke die geradlinige Fortsetzung der als Maßstab dienenden Teilung bildet."

Ein (Parallelverschiebungs-)Offset l_{off} zwischen Messstrecke und Maßverkörperung wirkt sich bei einer Verkippung α des Messsystems durch ein Führungsspiel als Längenmessfehler Δl aus und wird als Kipp- oder Abbefehler erster Ordnung bezeichnet.

$$\Delta l = l_{off} \cdot \sin(\alpha) \tag{2.2}$$

Da Verschiebungsoffset l_{off}, Kippwinkel α und Längenmessfehler Δl durch das Produkt in einer proportionalen Beziehung zueinander stehen, kann die Messabweichung durch die Minimierung von zwei Größen reduziert werden. Zum Einen kann der Offset durch eine fluchtende Anordnung von Messobjekt und Maßverkörperung reduziert werden.

$$l_{off} \to 0 \tag{2.3}$$

Damit können im Grenzfall die Abweichungen erster Ordnung vermieden werden. Parallel dazu kann zum Anderen das Führungsspiel minimiert werden, um auch Fehler zweiter Ordnung zu vermeiden.

$$\alpha \to 0 \tag{2.4}$$

Insbesondere die Abweichungen erster Ordnung haben bei einem Verschiebungsoffset l_{off} im Messsystem einen bedeutenden Einfluss auf dessen Messunsicherheit. Bereits eine Verkippung von $\alpha = 5$ " zieht bei einem Abbeoffset von $l_{off} = 100\,mm$ einen Längenmessfehler von $\Delta l = 2{,}42\,\mu m$ nach sich. Für das metrologische Grundgerüst einer NPM-Maschine ist es somit zwingend erforderlich, konstruktive Ansätze zur Minimierung von Abweichungen erster Ordnung zu realisieren und den Aufbau so zu gestalten, dass er dem Abbe'schen Komparatorprinzip entlang aller Raumachsen folgt. Die weitere Optimierung der Messunsicherheit kann durch die Vermeidung von Kippabweichungen zweiter Ordnung erreicht werden, indem die räumliche Orientierung des Messobjekts während einer Messung konstant gehalten wird. Dies ist durch geeignete Achsführungen und eine zusätzliche aktive Winkelregelung erreichbar. Da sich Abweichungen zweiter Ordnung nur gering auf die Messunsicherheit auswirken und die Erfassung und Regelung der räumlichen Orientierung des Positioniersystems einen erheblichen konstruktiven und messtechnischen Aufwand bedeutet, wird dieser Ansatz insbesondere bei Mikro-KMGs meist nicht verfolgt.

2.4.2.2 Aufbau von abbefehlerminimierten NPM-Maschinen

NPM-Maschinen sind komplexe mechatronische Systeme, welche aus einer Vielzahl mechanischer, optischer, elektronischer (analoger und digitaler) Teilsysteme bestehen. Da NPM-Maschinen im Kapitel 4 exemplarisch als nanometrologisches Positioniersystem für die Umsetzung des holistischen Steuerungsmodells zur Anwendung kommen, sollen ihr Aufbau und ihre Nutzerschnittstellen im Folgenden eingehend erläutert werden. Dazu wird zunächst der opto-mechanische Aufbau einer NPM-Maschine beschrieben und im Anschluss daran das Zusammenspiel aller Teilsysteme anhand eines Signallaufplans erläutert.

Der mechanische Aufbau von NPM-Maschinen besteht grundlegend aus dem metrologischen Rahmen einer frei positionierbaren Spiegelecke, welche zugleich als Träger für das Messobjekt dient, einem Tastsystem und mehreren Interferometern zur Detektion der Position der Spiegelecke. Im Gegensatz zu klassischen KMGs sind NPM-Maschinen als Moving-Stage-Systeme ausgeführt. Die in alle Raumachsen frei bewegliche Spiegelecke wird mit dem darauf befindlichen Messobjekt um das starre

Tastsystem herum bewegt [44], [46], [47], [48]. Die drei fest mit dem Rahmen verbundenen Interferometer sind entsprechend Abbildung 12 orthogonal zueinander angeordnet und auf die verspiegelten Außenflächen der Spiegelecke gerichtet. Durch die Reflexion der Laserstrahlen entstehen detektierbare Interferenzmuster, welche die Positionsbestimmung der Spiegelecke ermöglichen. Die Messstrahlen der Interferometer stellen somit die Längenmaßstäbe dar und bilden zusammen mit der Spiegelecke das intrinsische Koordinatensystem der NPM-Maschine. Die Verlängerungen ihrer Strahlachsen schneiden sich in einem Punkt, welcher die Position des Tastsystems definiert. Durch diese Anordnung ist das Abbe'sche Komparatorprinzip in allen drei Achsen erfüllt und Abbe'sche Messabweichung erster Ordnung können vollständig vermieden werden [46].

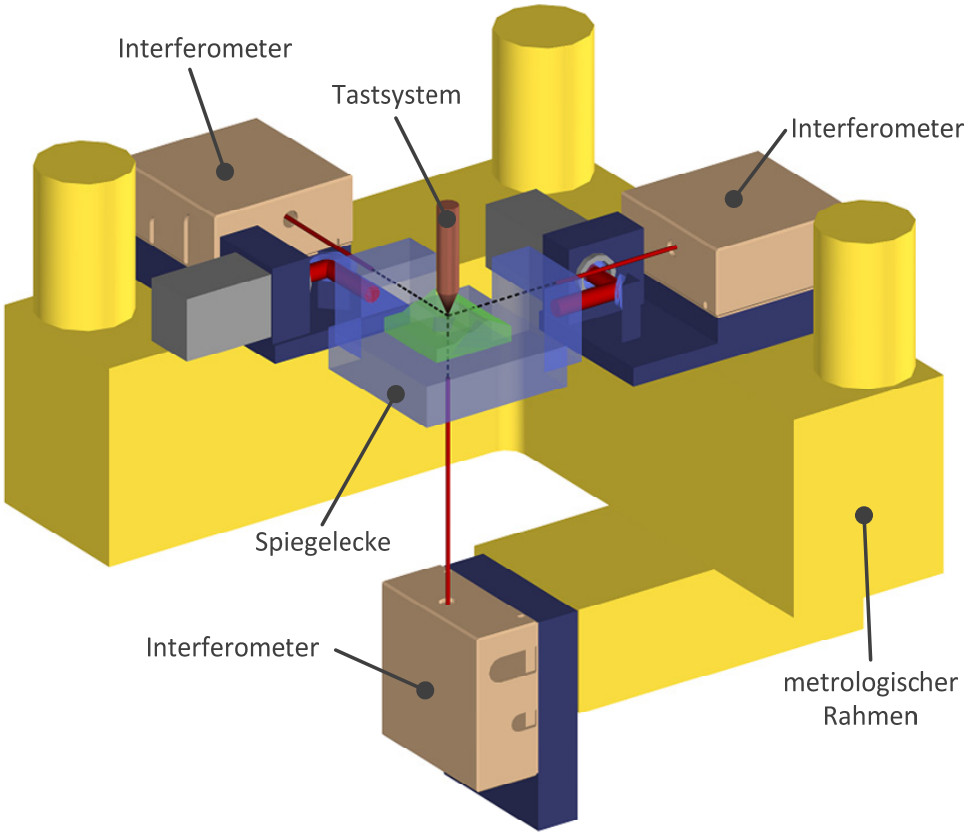


Abbildung 12: Prinzip einer NPM-Maschine [49]

Da es nach Manske [9] auch mit genauesten Justierverfahren schwierig ist, den Abbepunkt genauer als 0,5 mm zu justieren, ergibt sich in Verbindung mit der Unsicherheit der Führungen von 0,15 µm beim einem Stellbereich von 25 mm im Falle der Nanomessmaschine 1 ein Kippfehler von $\alpha = 1{,}24$ ", welcher nach Gleichung (2.2) einem Messfehler von $\Delta l = 3\,nm$ entspricht. Die weitere Reduzierung des Messfehlers ist nur durch eine Minimierung von Winkelabweichungen mittels aktiver Winkelregelung erreichbar. Dieser Ansatz wurde im Rahmen der Nanomesstechnik erstmals in NPM-Maschinen umgesetzt. Zur Bestimmung des Kippwinkels sind neben den drei Achsinterferometern zusätzliche Autokollimatoren erforderlich, mit welchen die Rotation der Spiegelecke um die drei Raumachsen bestimmt wird und über die Regelung der Antriebe ausgeglichen werden kann. So ist es möglich, den Kippwinkel auf $\alpha \leq 0{,}05$ " zu reduzieren, wodurch der Abbefehler auf $\Delta l \leq 0{,}1\,nm$ absinkt.

Der in Abbildung 13 dargestellte Signallaufplan visualisiert den erweiterten Aufbau von NPM-Maschinen unter Einbeziehung der zuvor aufgeführten Funktionsgruppen. Das zentrale Bauteil ist die digitale Datenverarbeitungseinheit. Sie zentralisiert und verarbeitet Messsignale, regelt die Antriebe,

koordiniert die Kommunikation zwischen Teilsystemen und stellt dem Anwender Datentransfer- und Messfunktionen über die Kommunikationsschnittstelle bereit. Veranlasst der Anwender das Verfahren der Spiegelecke, erfolgt eine Repositionierung der Spiegelecke über den Verstärker-Antriebs-Strang. Zeitgleich werden Position und Kippung der Spiegelecke über die digitalisierten Signale der Interferometer vom digitalen Regelungssystem überwacht und etwaige Fehler ausgeglichen. Auch das Tastsystem wird von der zentralen Datenverarbeitungseinheit über die zugehörigen Signalwandler ausgelesen. Diese Daten werden zum Einen im Rahmen einer Messung dem Anwender bereitgestellt und zum Anderen werden sie für die Überwachung des Tastsystems verwendet, um im Fall von unvorhergesehenen Kollisionen zwischen Tastsystem und Messobjekt die Antriebe zu stoppen und so eine Beschädigung des Tastsystems zu vermeiden.

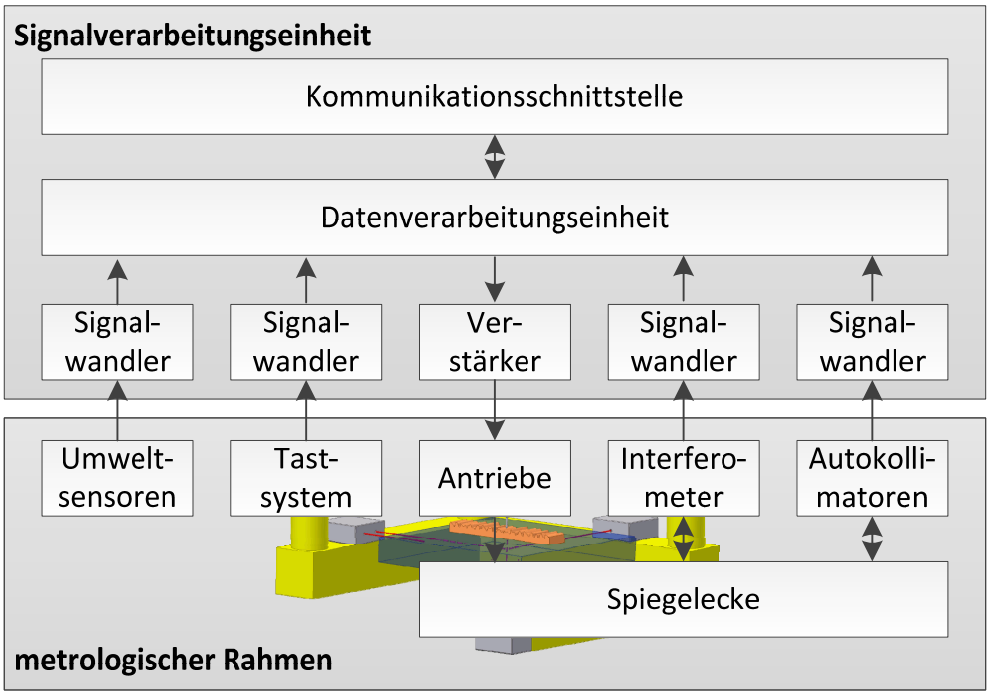


Abbildung 13: Funktionskomponenten von NPM-Maschinen

2.4.2.3 Nanomessmaschine NMM-1

Die Nanomessmaschine NMM-1 (kurz: NMM-1, siehe Abbildung 14) stellt die erste Generation von NPM-Maschinen dar. Sie setzt das Funktionsprinzip von NPM-Maschinen vollständig um und ermöglicht in ihrem Messvolumen von 25 x 25 x 5 mm^3 eine Positionierauflösung von 0,1 nm. Durch die Umsetzung des Abbe-Komparatorprinzips kann in Kombination mit der Winkelregelung eine Positionierunsicherheit von unter 10 nm erzielt werden [50]. Die NMM-1 erlaubt eine maximale Samplefrequenz von 6,25 kHz. Die zentrale Datenverarbeitungseinheit der NMM-1 ist ein digitaler Signalprozessor, welcher für die Vorverarbeitung der Rohmessdaten auf mehrere FPGA-Module zurückgreift. Die Steuerung einer NMM-1 erfolgt hardwareseitig über eine USB-Schnittstelle.

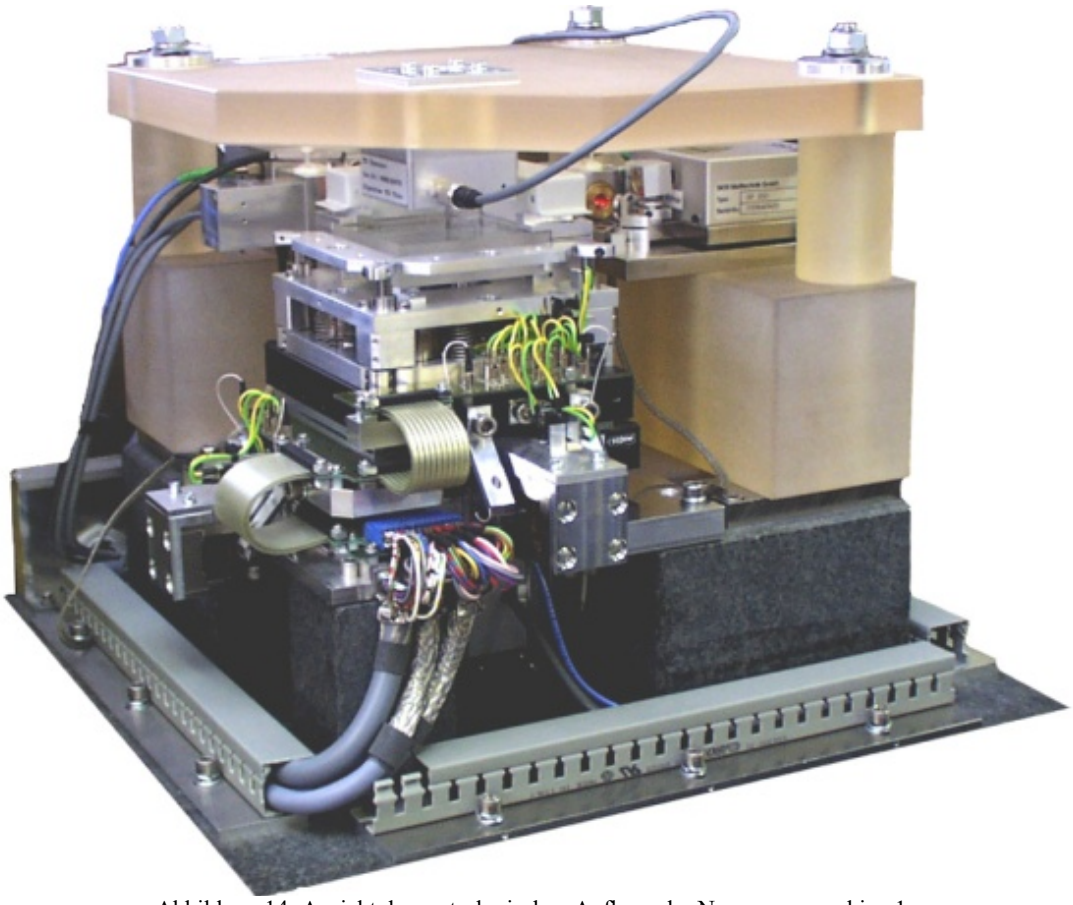


Abbildung 14: Ansicht des metrologischen Aufbaus der Nanomessmaschine 1

Nutzerschnittstelle der NMM-1

Der Datenaustausch zwischen Anwender und NMM-1 laufen über eine Treiberkaskade ab, welche über mehrere Funktionsbibliotheken und den USB-Treiber mit der NMM-1-Hardware kommunizieren. Durch diese Treiberkaskade werden die komplexen Kommunikationsmechanismen transparent gekapselt. Da es über die USB-Schnittstelle nicht möglich ist, direkt auf den Firmwarecode der NMM-1 zuzugreifen, müssen alle Aufrufe mit den dafür notwendigen Parametern nutzerseitig als Datenpaket verpackt und über die USB-Schnittstelle an die NMM-1 versendet werden. Dieser Mechanismus ist eine immer wiederkehrende Aufgabe, welche von der Treiberkaskade übernommen wird. Das Top-level der Treiberkaskade (siehe Abbildung 15) stellt dem Nutzer alle NMM-1-Funktionen für Messung und Parametrierung als ANSI-C Funktionen bereit. Jeder Aufruf dieser Funktionen zieht eine Parameterprüfung nach sich, bevor aus den Aufrufparametern automatisch Aufrufpakete erstellt und an die NMM-1 versendet werden. Von der NMM-1 empfangene Pakete werden auf syntaktische und logische Korrektheit überprüft und ausschließlich nach bestandener Prüfung als Antwortparameter oder Messdaten an den Anwender übergeben.

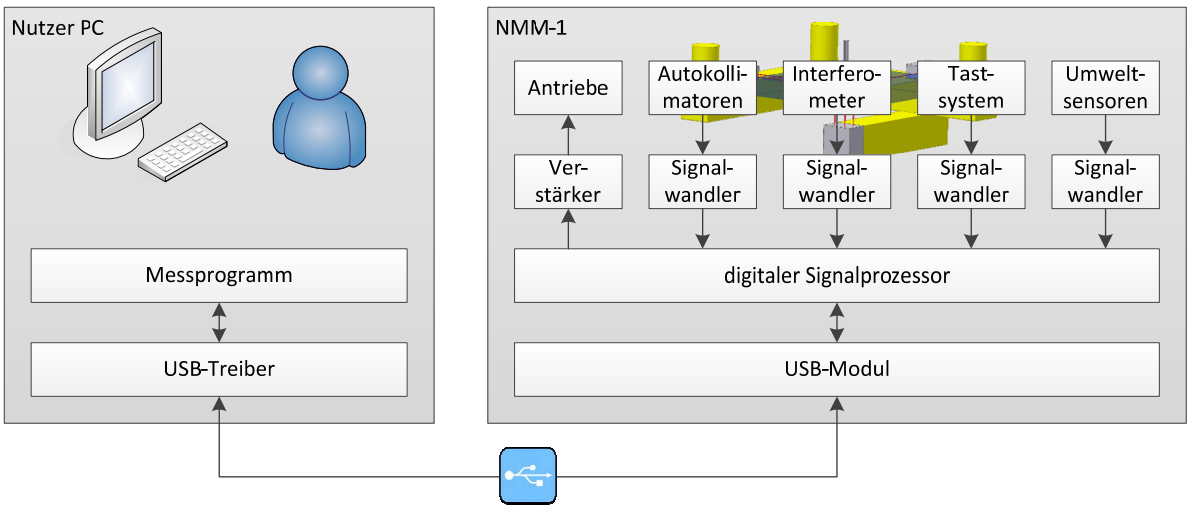


Abbildung 15: Datenströme bei einer Nanomessmaschine 1

Steuerung der NMM-1

Die Steuerung einer NMM-1 erfolgt durch das Einbinden ihrer USB-Treiber in ein Anwenderprogramm. Üblicherweise wird dafür die Entwicklungsumgebung MathWorks Matlab verwendet. Mit Matlab ist es vergleichsweise einfach, die Bibliothek zu laden und über ein textbasiertes Messskript anzusprechen. Für das Erstellen der Messskripte sind jedoch gute Programmierkenntnisse und umfangreiches Wissen über die NPM-Maschine erforderlich. Für jede Messaufgabe muss mit dem Matlab-Texteditor ein neues Messskript erstellt werden, welches den kompletten Ablauf der Messung inklusive der Datenübertragung und Verarbeitung bewerkstelligen muss. Das Fehlen standardisierter Datenverarbeitungsverfahren erfordert, dass die Auswertung der gewonnenen Messdaten für jedes Skript problemspezifisch neu erstellt werden muss, was den zeitlichen Aufwand für die Erstellung eines Skriptes erhöht. Mit wachsender Komplexität von Messobjekten und den zu analysierenden Eigenschaften stößt diese Herangehensweise an ihre Grenzen. Es ist somit wünschenswert, den Anwender bei der Erstellung von Messskripten durch grafische Werkzeuge und standardisierte Datenverarbeitungsverfahren zu unterstützen.

2.4.2.4 NPMM-200

Parallel zu den Arbeiten an der NMM-1 wird eine neue Generation - die NPMM-200 - mit deutlich größerem Messvolumen, höherer Messfrequenz und verbessert Positionierauflösung entwickelt [51] (Parameter der NPMM-200 siehe Tabelle 6).

Parameter	Wert
Positionierauflösung	0,08 nm
Positionierunsicherheit	1 nm
max. Abtastfrequenz	83,33 kHz
Messvolumen	200 x 200 x 50 mm³

Tabelle 6: Parameter von NPMM-200 **[51]**

Wie die NMM-1 folgt auch der mechanische Aufbau einer NPMM-200 den Grundprinzipen zum Aufbau von NPM-Maschinen. Da die Samplefrequenz der NPMM-200 im Vergleich zur NMM-1 um über das 13-fache gesteigert wurde, ist es mit einer monolithischen Datenverarbeitungseinheit nicht mehr möglich, alle Datenverarbeitungsschritte in hinreichend kurzer Zeit zu bewerkstelligen. Die gesteigerte Samplefrequenz ist dabei nur ein Teilaspekt. Um die Positionierunsicherheit zu reduzieren, kommen neuartige, komplexere Modelle zur Beschreibung von Reibprozessen in den

Antriebsführungen zum Einsatz, welche eines entsprechend hohen Datenverarbeitungsaufwands bedürfen. Aus diesem Grund ist die Datenverarbeitung bei der NPMM-200 auf ein heterogenes, dreigeteiltes Rechnersystem verteilt. Den drei zentralen Rechnersystemen sind jeweils spezifische Aufgaben zugeordnet. Entsprechend der Erfordernisse der jeweiligen Datenverarbeitungsalgorithmen der Teilsysteme verfügen diese zusätzlich über untergeordnete Datenverarbeitungseinheiten. So nutzt das Data Acquisition System (kurz: DAS) für die schnelle Erfassung und Vorverarbeitung der gemessenen Daten mehrere, parallel arbeitende Streamprozessoren, um das hohe Datenaufkommen bewältigen zu können.

NPMM 200 Teilsystem	Aufgabe
Sequence Control System (SCS)	Ablaufsteuerung und Koordinierung der Subsysteme
Control System (CS)	Positionsregelung und Bahnplanung
Data Acquisition System (DAS)	Messdatenerfassung und Messdatenvorverarbeitung

Tabelle 7: Teilsysteme der NPMM 200

Nutzerschnittstelle der NPMM-200

Die zentrale Steuereinheit der NPMM-200, das Sequence Control System (kurz: SCS), stellt dem Anwender einen mit der NMM-1 abwärtskompatiblen Befehlssatz bereit. Erweiterungen sind dahingehend erfolgt, dass die NPMM-200 deutlich mehr Umwelt- und Systemdaten aufzeichnet und diese dem Anwender zugänglich macht. Ferner wurde die Nutzerschnittstelle der NPMM-200 von einem hardwarespezifischen Übertragungsmedium entkoppelt und auf unabhängige TCP als Datenübertragungsgrundlage festgelegt. Aufgrund des potentiell hohen Datenaufkommens der NPMM-200 wurde ein besonderes Augenmerk auf eine Maximierung der Datentransferrate der Nutzerschnittstelle gelegt. Fast alle gängigen Ethernet-Netzwerkkarten unterstützen derzeit Bruttoübertragungsraten von bis zu einem Gigabit pro Sekunde, was mehr als die doppelte Transferrate der NMM-1 ist. Die Abtrennung von einem spezifischen Medium mittels TCP ermöglicht zukünftig die Anwendung innovativer Übertragungsverfahren mit deutlich höheren Transferraten. Das verbesserte Datenübertragungsprotokoll integriert vergleichbar zu I++DME eine asynchrone Kommunikation und einen Mechanismus zur Ereignisübermittlung, wodurch die Transferlast zusätzlich vermindert werden kann. Durch die Ereignisübermittlung sendet jedes der drei Rechnersysteme eigenständig Messdaten, Fehler und Zustandsänderungen (siehe Abbildung 16) über einen virtuellen Übertragungskanal an den Nutzer.

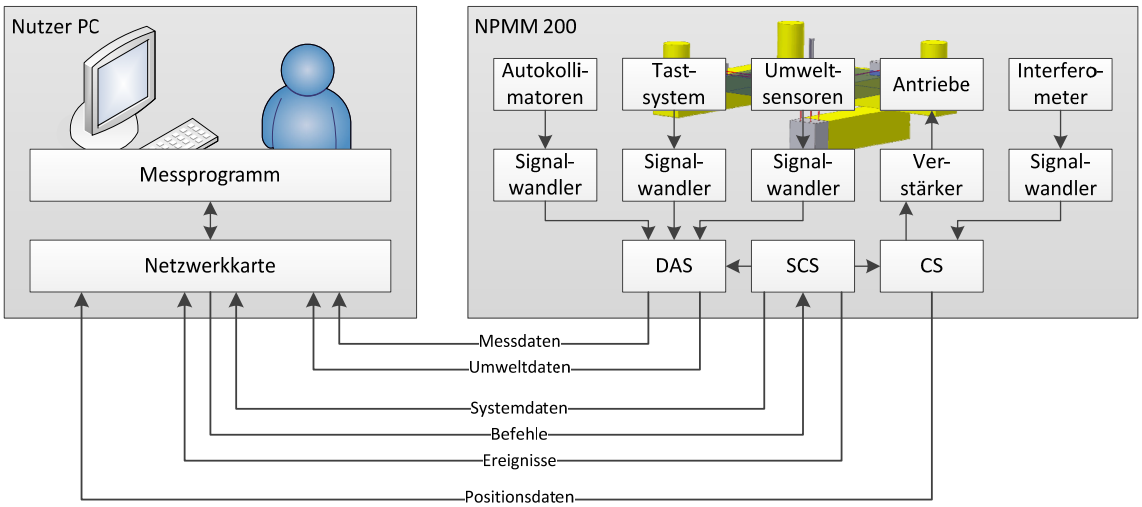


Abbildung 16: Datenströme bei einer NPMM-200

Steuerung der NPMM-200

Die Vielzahl der Datenkanäle und die damit verbundene Komplexität bei der Steuerung der NPMM-200 und der Auswertung von Messdaten erfordert einen ungleich höheren Kenntnisstand über die Funktionsweise der NPMM-200 als bei der NMM-1. Wie im Falle der NMM-1 ist der Aufwand für die Erstellung textbasierter Messskripte in Matlab hoch und die gewonnenen Daten bedürfen aufgrund ihres asynchronen Eintreffens über die Vielzahl der Übertragungskanäle einer drastisch gesteigerten Verarbeitungskomplexität. Darüber hinaus erschweren die wachsenden Anforderungen zur Messung komplexer Objektstrukturen zusätzlich die Erstellung der textbasierten Messskripte, sodass auch in Verbindung mit der NPMM-200 eine Alternative mit hohem Grad der Nutzerunterstützung dem bisherigen Steuerungskonzept vorzuziehen ist.

Diese Lücke soll durch die vorliegende Arbeit geschlossen werden. Durch die Erarbeitung und Integration des transparenten, holistischen Steuerungskonzepts soll die hohe Komplexität bei der direkten Kommunikation mit einer NPMM-200 vor dem Anwender verborgen und die Steuerung der NPM-Maschine über eine einfach zu handhabende Schnittstelle, wie z. B. I++DME, erreicht werden. Durch die Anwendung einer etablierten und allgemein akzeptierten Schnittstelle für die Nanomesstechnik entsteht die Möglichkeit, NPM-Maschinen mit ausgereifter, kommerzieller und intuitiv bedienbarer Anwendersoftware zu steuern. Um diese Schnittstellen für nanometrologische Anwendungen nutzen zu können, muss zunächst untersucht werden, inwieweit sie für die Lösung nanomesstechnischer Herausforderungen geeignet sind oder ob gegebenenfalls Erweiterungen im Sinne der Nanometrologie erforderlich werden. Die Ergebnisse umfassender Untersuchungen bezüglich dieser Fragestellung sind im Kapitel 4.2 dargelegt.

2.5 Sensoren für nanometrologische Messanwendungen

Ebenso wie makroskopische KMGs benötigen auch NPM-Maschinen Sensoren, um einen Bezug zwischen dem Messobjekt und den Längenmaßstäben der Positionierachsen herzustellen. Um im Rahmen einer Antastung objektbeschreibende Raumkoordinaten zu speichern, werden von den Antastsystemen verschiedene Methoden zur Detektion der Objektoberfläche umgesetzt. Die Bezugnahme erfolgt bei taktiler Antastung durch die Erzeugung einer Antastkraft oder die Auswertung photometrischer Größen bei den optischen Sensoren. Taktile Sensoren können entsprechend der Dimensionalität ihres Detektorsystems weiter in 2,5D-Sensoren oder 3D-Sensoren untergliedert werden. Im Gegensatz zu klassischen KMGs wird bei NPM-Maschinen eine Vielzahl

optischer Messverfahren eingesetzt. Die optischen Sensoren ermöglichen das berührungslose Antasten von Oberflächen und sind somit insbesondere für empfindliche, weiche und druckempfindliche Messobjekte interessant. Durch den entsprechenden Aufbau optischer Sensoren können im Rahmen einer Messung unter Umständen auch eine Vielzahl von Objektpunkten gleichzeitig angetastet werden, wodurch die Dauer für die Datenerfassung signifikant gesenkt werden kann.

Die in den Abschnitten 2.2 bis 2.4 erläuterten Sensortypen und deren Messprinzipien können für die Anwendung in NPM-Maschinen als Sensorsysteme entsprechend der Abbildung 17 systematisiert werden. Die grundlegende Untergliederung erfolgt in taktile und optische Sensoren, welche weiter nach ihrer 3D-Messfähigkeit und Parallelisierbarkeit klassifiziert werden.

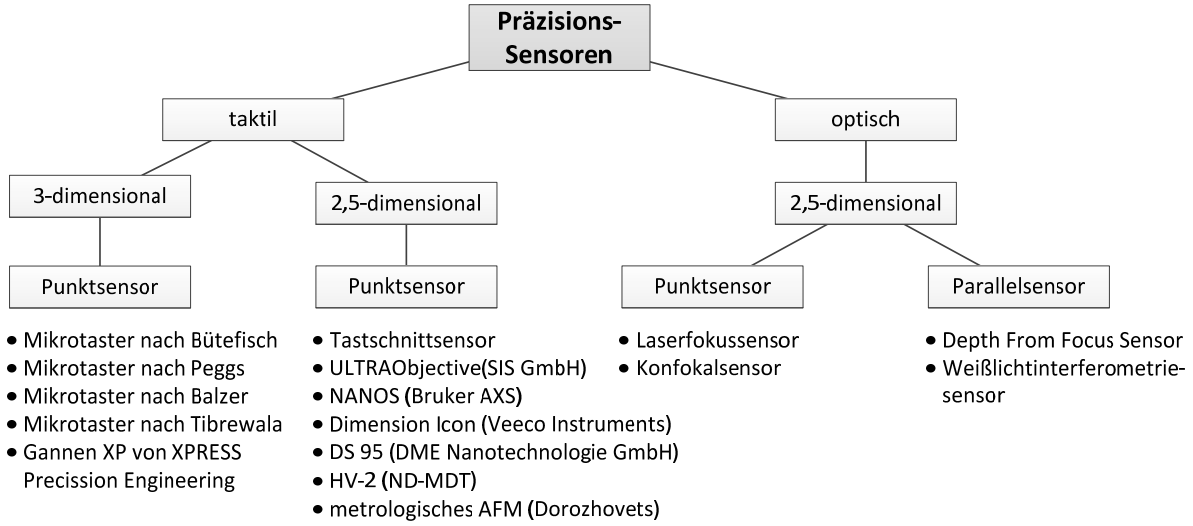


Abbildung 17: Klassifikation von Sensoren für NPM-Maschinen

2.5.1 Taktile Mikrosensoren

In verschiedenen Projekten, insbesondere innerhalb des Sonderforschungsbereiches 622 wurde eine Vielzahl von taktilen Sensorsystemen für NPM-Maschinen entwickelt und Präzisionssensoren dritter in NPM-Maschinen integriert. Darunter zählen verschiedene 3D-Mikrotaster [52], [53], [54], [55], [56], [57], welche auf verschiedene Weise das Prinzip des messenden Sensors realisieren und für die schnelle Datenerfassung mit Scanmessungen geeignet sind. In Abbildung 18 ist der Einsatz eines Mikrotaster nach Bütefisch [55] (siehe auch Abbildung 19 b) dargestellt. Im Gegensatz zu den mehrere Millimeter betragenden Querschnitten makroskopischer Sensoren besitzen Mikrotaster Tastkugeln mit Durchmessern zwischen 300 µm und 70 µm. Der kleine Durchmesser hat nach Hertz [58] zur Folge, dass sich die Kontaktfläche in Bezug auf die Radiusverringerung quadratisch verringert, weshalb auch die Antastkräfte entsprechend verringert werden müssen, um eine plastische Verformung von Messobjekt bzw. Tastkugel zu verhindern. Darüber hinaus realisieren Mikrotaster nanometrologischer Eignung Unsicherheiten von wenigen Nanometern. Die Detektion der Auslenkung erfolgt basierend auf verschiedenen Wirkprinzipien durch Weg- oder Biegungssensoren an einem planaren Balkenfedersystem (siehe Abbildung 19). Durch diese Anordnung besitzt das Detektorsystem oft anisotrope Federkonstanten, weshalb die Messunsicherheit der Sensoren von der Antastrichtung abhängig ist.

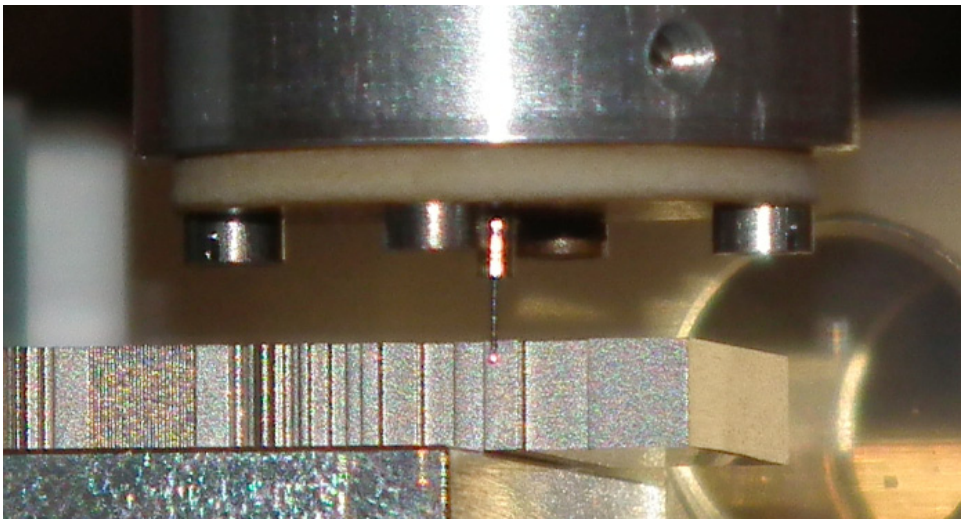
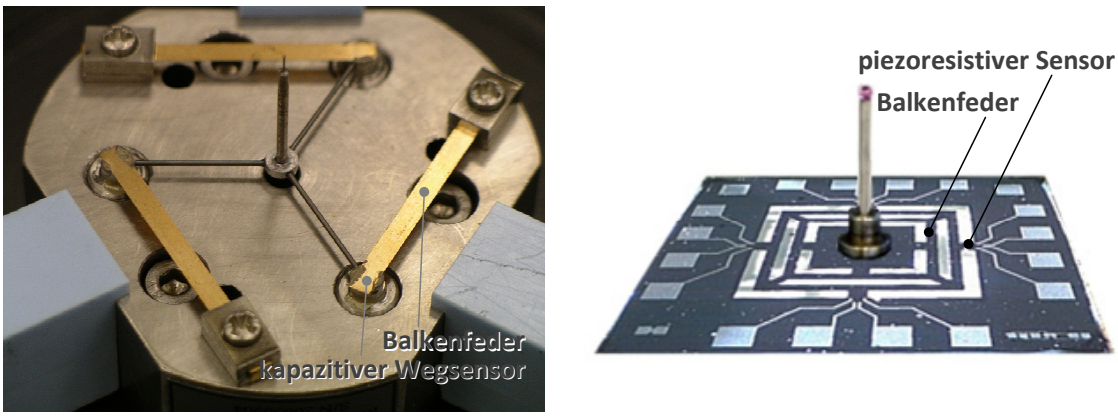


Abbildung 18: Taktile Messung mit einem Mikrotaster an einer NMM-1



(a) IBS-NPL 3D-Mikrotaster (b) IMT 3D-Siliziummembrantaster

Abbildung 19: Aufbau zweier Mikrotaster

Als weitere taktile Mikro- und Nanosensoren wurden neben einem Tastschnittsensor [59] weiterhin eine Vielzahl kommerzieller AFM-Systeme in NPM-Maschinen integriert (siehe Tabelle 8 und Abbildung 20).

AFM-System	Hersteller/Institut
ULTRAObjective	SIS GmbH
NANOS	Bruker AXS
Dimension Icon	Veeco Instruments
DS 95	DME Nanotechnologie GmbH
metrologisches AFM	TU Ilmenau

Tabelle 8: AFM-Sensoren für NPM-Maschinen

Das metrologische AFM nach Dorozhovets [60] ist eine Entwicklung für NPM-Maschinen im Rahmen des SFB 622. Das AFM verfügt im Gegensatz zu kommerziellen AFMs lediglich über ein piezoelektrisches Positioniersystem für die z-Achse und die Schwingungsanregung des Biegebalkens, da die räumliche Positionierung des Sensors über die NPM-Maschine erfolgt. Sowohl Tastschnitt- als auch AFM-Sensor sind eindimensional messende Systeme und sind im Gegensatz zu 3D-Mikrotastern nur zur Erfassung von 2,5D Daten befähigt. Aufgrund des applizierten Messprinzips und ihrer geometrischen Parameter (siehe Tabelle 9) sind AFM-Sensoren im Gegensatz zu Mikrotastern befähigt höhere Strukturauflösungen bei einer geringeren Messunsicherheit zu erreichen.

Abbildung 20: AFM-Sensor HV-2 von ND-MDT an einer NMM-1

Parameter	3D-Mikrotastern	AFM-Sensoren
Auflösung [nm]	3..1	< 1
Dynamikbereich [µm]	100..20	< 0,1
Unsicherheit [nm]	500..5	3
Tastkugelradius	1..0,15 mm	≥ 10 nm

Tabelle 9: Parameter von taktilen Sensoren (Quelle: **[61]**)

Sowohl die AFM-Sensoren als auch die 3D-Mikrotaster sollen im folgenden Kapitel in den Entwurf des holistischen Steuerungskonzepts für NPM-Maschinen aufgenommen werden. Beide Sensorklassen bilden neben den optischen Sensoren das Grundgerüst für die Messdatenerfassung. Aufgrund ihrer unterschiedlichen Messverfahren und der unterschiedlichen Eignung zum 3D-Messen bedürfen die Klassen unterschiedlicher Antastverfahren, deren Bereitstellung durch das Steuerungsmodell realisiert werden muss. Als exemplarische Implementierung wird der Bütefisch-Mikrotaster im Kapitel 3 als taktiles Messsystem basierend auf dem holistischen Steuerungskonzept für die NPM-Maschinen integriert und in Kapitel 6 für experimentelle Untersuchungen an der Softwareimplementierung zur Anwendung gebracht.

2.5.2 Optische Präzisionssensoren

Die zweite Sensorgruppe für NPM-Maschinen basiert auf optischen Messverfahren. Optische KMG-Sensoren, von denen die meisten auf dem Triangulationsverfahren basieren, erlauben eine sehr schnelle Datenerfassung mit hinreichend geringen Unsicherheiten für makroskopische Anwendungen. Zur Nutzung für nanometrologische Zwecke weisen diese Sensoren in der Regel jedoch eine zu hohe Messunsicherheit auf. Die in NPM-Maschinen eingesetzten optischen Sensorsysteme wurden mit besonderer Rücksicht auf die Verbesserung ihrer metrologischen Eigenschaften im Rahmen des SFB 622 entwickelt. So verfügen konventionelle Laserfokussensoren über Unsicherheiten von über 10 nm, wohingegen der Fokussensor für den Einsatz in NPM-Maschinen [15] eine Messunsicherheit von weniger als 1 nm aufweist. Als optisch messende Arraysensoren wurden ein Weißlichtinterferenzmikroskopsensor (WLI) [62] und ein Depth-From-Focus-Sensor (DFF) [18]

entwickelt. Der Aufbau beider Sensorsysteme ist bis auf das verwendete Objektiv identisch (siehe Abbildung 21). Aufgrund seines Messprinzips erfordert der WLI-Sensor die Nutzung eines Mirau-Interferenzobjektivs, wohingegen der DFF-Sensor mit einem konventionellen Mikroskopobjektiv ausgestattet ist. Die Sensorsysteme unterscheiden sich neben der Möglichkeit zur parallelen Datenerfassung vor allem in der erreichbaren Auflösung und dem möglichen Dynamikbereich (siehe Tabelle 10).

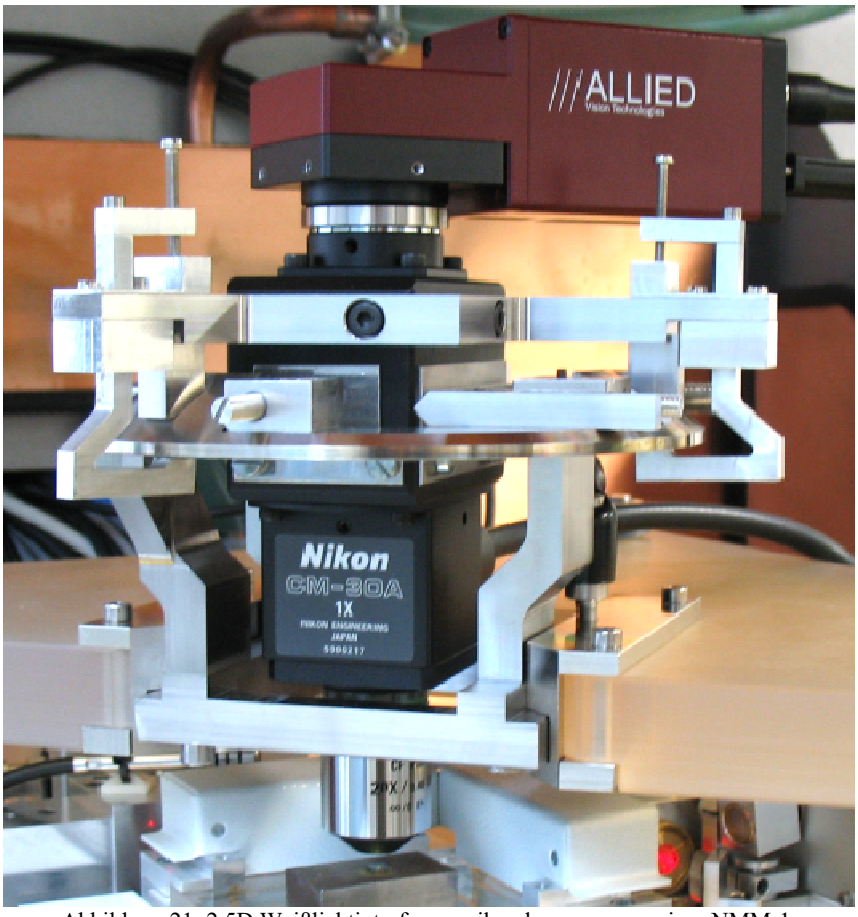


Abbildung 21: 2,5D Weißlichtinterferenzmikroskopsensor an einer NMM-1

Auch die optischen Sensoren sollen im folgenden Kapitel in das Steuerungsmodell für NPM-Maschinen aufgenommen werden. Stellvertretend für diese Klasse wird im Kapitel 5 ein Weißlichtinterferenzmikroskopsensor basierend auf dem holistischen Steuerungskonzept in die Steuerungssoftware aus Kapitel 3 eingebunden. Im Kapitel 6 werden anschließend in Verbindung mit einer NPM-Maschine als Positioniersystem experimentelle Untersuchungen mit diesem optischen Sensor durchgeführt und die Ergebnisse dargestellt und analysiert.

Parameter	Laserfokus	WLI	DFF
Auflösung [nm]	< 0,1	< 1	50
Dynamikbereich [µm]	3	300	300
Unsicherheit [nm]	ca. 5	-	-

Tabelle 10: Parameter von optischen Sensoren

2.6 Metrologische Erweiterungen für NPM-Maschinen

Seit der Gründung des Sonderforschungsbereichs 622 „Nanopositionier- und Nanomessmaschine“ in 2002 wurde die NMM-1 von den Wissenschaftlern der TU Ilmenau in enger Kooperation mit der SIOS Messtechnik GmbH permanent weiterentwickelt. Aus der langjährigen Forschung an NPM-Maschinen entstand neben den Sensoren ein System zur Gewichtskraftkompensation [63], welches die vertikalen Antriebe entlastet und die Messunsicherheit durch den verringerten Wärmeeintrag senken kann.

Die aktuellen wissenschaftlichen Bestrebungen verfolgen das Ziel, NPM-Maschinen im Sinne der iMERA Technology Roadmap [8] weiterzuentwickeln. Insbesondere ist es von Interesse, das Multisensorkonzept für nanometrologische Applikationen zu realisieren. Nur so können die Vorteile verschiedener Messverfahren kombiniert werden [9]. Eine Sensorwechseleinrichtung für optische und taktile Sensoren auf Basis eines Mikroskoprevolvers ist in Abbildung 22 dargestellt. Die Sensoren können für verschiedene Analysen am Messobjekt ohne nennenswerten Eingriff in das System durch Drehen des Revolvers getauscht werden.

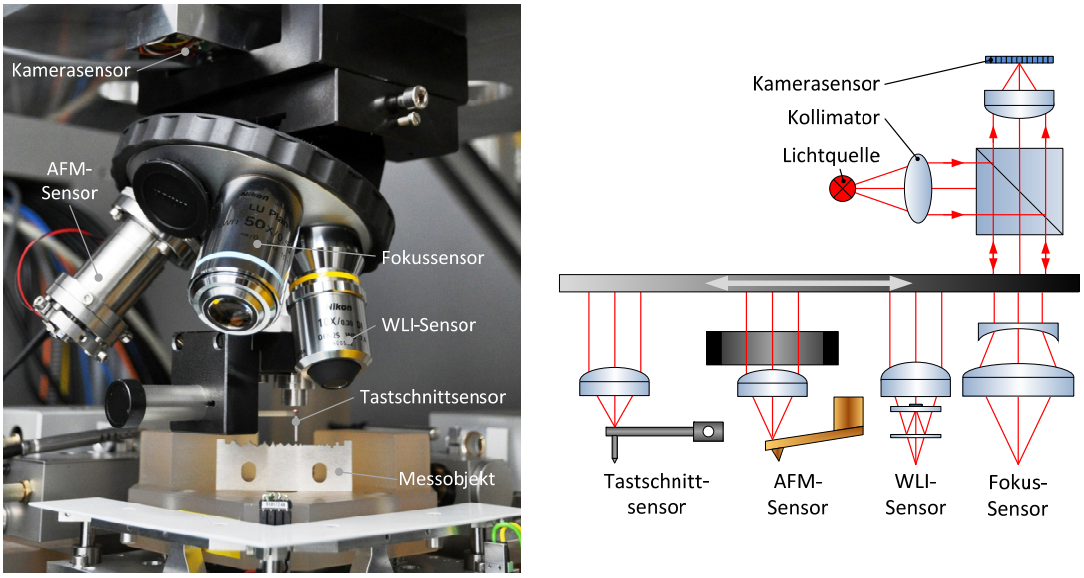


Abbildung 22: Sensorrevolver mit verschiedenen Sensoren an einer NMM-1

Um die fehlende Fähigkeit der Messung an Hinterschneidungen bei eindimensional messenden Sensoren wie Tastschnittsensor oder optischen Sensoren zu kompensieren, wurde der in Abbildung 23 dargestellte, miniaturisierte zweiachsige Kipp- und Drehtisch entwickelt. Durch das Drehen bzw. Kippen wird das Messobjekt relativ zur Messachse des Sensors so angeordnet, dass dieser die Hinterschneidung vermessen kann bzw. die durch die Tastkörperform taktiler Sensoren hervorgerufenen morphologischen Effekte an steilen Kanten ausgeschlossen werden können. Weckenmann schlägt im Falle der Antastung steiler Anstiege vor, den Sensor mithilfe einer Sensorkippeinrichtung orthogonal zur fraglichen Kante auszurichten [64]. Da jedoch nur die relative Ausrichtung von Sensor und Objekt relevant ist, kann statt des Sensors auch das Messobjekt ausgerichtet werden, um die morphologischen Effekte zu umgehen. Für kleine Messobjekte mit einem Volumen im Bereich weniger Kubikzentimeter ist der zweiachsige rotierbare Tisch eine geeignete Erweiterung für NPM-Maschinen [65].

Die zusätzlich auf dem Kipptisch angebrachten Rubinkugeln dienen dazu, den exakten Bezug zum inerten Koordinatensystem der NPM-Maschine nach einer Rotation bzw. Kippung mittels Einmessen dieser Zielelemente herzustellen. Da die Führung des Tisches nur mit endlicher Präzision gefertigt

werden kann, ist sie nicht hinreichend präzise, um eine Translation bzw. Rotation aufgrund von nicht idealen Lagern vernachlässigen zu können.

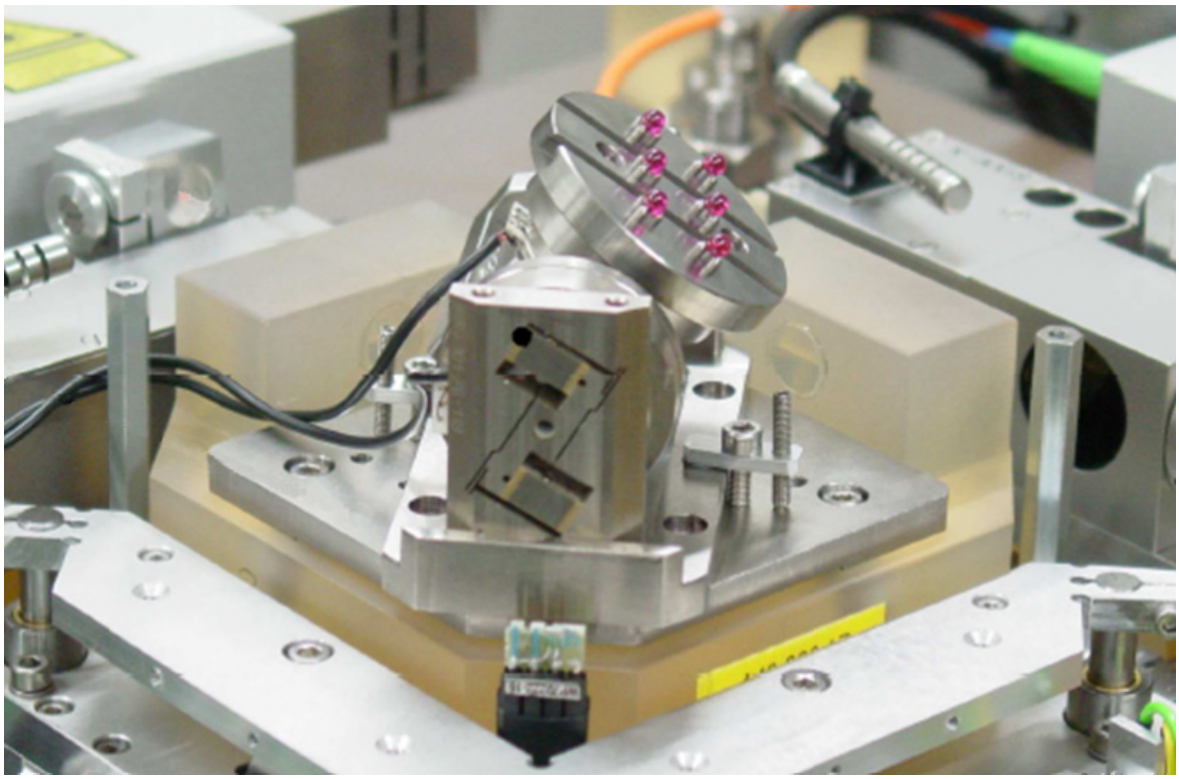


Abbildung 23: Kipp- und Drehtisch für NPM-Maschinen mit Bezugselementen

Nicht zuletzt soll auch die Möglichkeit der Mikro- und Nanobearbeitung von Oberflächen Berücksichtigung finden. Oeder stellt in zwei Veröffentlichungen das Prinzip [66] und das Design [67] einer optischen Pinzette vor. Die optische Pinzette ist ein Werkzeug, welches zum Einfangen von Mikro- und Nanopartikeln verwendet wird. Zur punktuellen Bearbeitung von Oberflächen (Nanolithografie) kommen meist Laserwerkzeuge oder spezielle Rasterkraftsonden zum Einsatz [68]. Mit beiden Verfahren ist es möglich, eine Objektoberfläche in atomaren Dimensionen zu manipulieren - beim Laser durch hochenergetische Strahlung, bei Rasterkraftsonden durch das Anlegen von Spannungen oder mechanisches Einwirken. Ferner existieren auch zahlreiche Methoden, durch welche Partikel mit einer Rasterkraftsonde auf Oberflächen abgeschieden werden [69], um so gezielt Nanostrukturen zu erzeugen.

Eine mit Sensorwechsler, Kipp-/Drehtisch bzw. Nanotools ausgestattete NPM-Maschine erlaubt im Vergleich zu einem konventionellen Gerät bisher nicht realisierbare nanometrologische Untersuchungen und stellt desweiteren einen bedeutenden Schritt zur Schaffung einer visionären NPM-Maschine im Sinne der iMERA [9] dar. Durch die Unterstützung von Sensor-Wechsel-Einrichtungen können kombinierte und hoch komplexe Analysen vorgenommen werden, welche im Rahmen einer multisensoriellen Messung verschiedene Eigenschaften des Messobjekts analysieren und die heterogenen Messdaten zu einem Gesamtergebnis fusionieren. Mit Kipp-/Drehtischen kann das Messobjekt frei im Raum rotiert werden, so dass durch die Unterstützung entsprechender Datenfusionsalgorithmen auch 2,5D-Sensoren befähigt werden, in Hinterschneidungen zu messen. Nicht zuletzt eröffnet die Integration von Nanotools die Möglichkeit, ein Objekt lokal auf der atomaren Ebene zu manipulieren. So gelangen neben neuartigen Analyseverfahren zukunftsweisende Ansätze im Sinne der Nanofabrikation in den Fokus der Nanotechnologie. Damit können NPM-Maschinen perspektivisch von einem Messgerät zu einem Instrument für die Synthese von Mikro- und Nanobauteilen transformiert werden. Diesbezüglich sind jedoch noch viele technische

Herausforderungen zu lösen. Die vorliegende Arbeit stellt dabei einen ersten Schritt dar, um die heterogenen nanometrologischen Einzelsysteme zu einer komplexen, hoch flexiblen NPM-Maschine im Sinne der iMERA zu vereinen. Die etablierten Ansätze zum Aufbau von Steuerungssoftware für NPM-Maschinen sind jedoch zu starr, um diesem Anspruch gerecht zu werden. Deswegen muss vielmehr ein neuartiges, flexibles und modulares Konzept entwickelt werden, welches die optimale Skalierbarkeit des Gesamtsystems ermöglicht. Ein derartiges Modell wird im folgenden Kapitel erarbeitet und im weiteren Verlauf der Arbeit am Beispiel von NPM-Maschinen sukzessive zu einem Demonstrator ausgebaut.

Kapitel 3 Entwurf des Steuerungskonzepts

Im vorigen Kapitel wurde das heterogene technologische Umfeld der Nanomesstechnik dargelegt. Verschiedene sensorische Verfahren aus Mikroskopie, Rastersondenmikroskopie und Koordinatenmesstechnik stehen Seite an Seite mit Positionier- und Messsystemen unterschiedlicher konstruktiver und messtechnischer Ansätze. Zwar sind generell alle vorgestellten Systeme für metrologische Untersuchungen im Bereich zwischen der makroskopischen und der atomaren Skala qualifiziert, jedoch sind sie in Abhängigkeit der metrologischen Aufgabenstellungen, der Strukturierung des Messobjekts bzw. seiner Materialeigenschaften für die Lösung einer Aufgabe unterschiedlich gut geeignet. Die einfache, transparente und umfassende Kombinationsmöglichkeit aller Systeme würde neue Möglichkeiten für die Nanometrologie eröffnen und ihren methodischen Umfang maßgeblich erweitern.

Gerade an diesem Punkt tritt das Problem fehlender Interoperabilität der nanometrologischen Teilsysteme deutlich zu Tage. Die gegenwärtig fehlende Möglichkeit, essentielle Programmmodule zur Steuerung der einzelnen Hardwarekomponenten als strukturierte Software zu kombinieren bildet eine enorme Hürde für die Weiterentwicklung der Nanometrologie. Dieses Hindernis soll im Rahmen der vorliegenden Arbeit überwunden werden. Dazu soll im Folgenden ein abstraktes, umfassendes und allgemeingültiges Modell für nanometrologische Messgeräte eingeführt werden. Im Verlauf dieses Kapitels werden diesbezüglich zunächst die applikativen Anforderungen an ein Steuerungskonzept gemäß der iMERA [70] zusammengefasst, um daraus nachfolgend ein holistisches Steuerungskonzept

für nanometrologische Messsysteme zu erarbeiten, welches die gesetzte Zielstellung realisiert und hinreichend flexibel gestaltet ist, um auch zukünftige Entwicklungen konsistent zu integrieren.

3.1 Anforderungen an das Steuerungsmodell für NPM-Maschinen

Der Entwurf des Steuerungsmodells für NPM-Maschinen wird im Folgenden basierend auf den in Abbildung 24 abgebildeten Anwendungsfällen für eine NPM-Maschine erarbeitet. Der Aufbau von NPM-Maschinen wird zunächst nicht berücksichtigt, sondern vielmehr die prinzipiellen Anforderungen erfasst. Aus Anwendersicht müssen NPM-Maschinen Möglichkeiten zum Anpassen und persistenten Speichern von Einstellungsparametern des Messsystems bereitstellen. Darin inbegriffen sind auch Daten, welche im Rahmen der Sensorkalibrierung gewonnen werden. Da die Interaktion mit einem Messobjekt von zentraler Bedeutung des Messsystems ist, beziehen sich die übrigen Anwendungsfälle auf die Interaktion mit selbigem. Das Objekt muss vom Anwender entsprechend der Messaufgabe frei im von der NPM-Maschinen bereitgestellten Messvolumen verschoben, gekippt und rotiert werden, um Koordinaten- oder Rastermessungen mit taktilen oder optischen Sensoren vornehmen zu können. Darüber hinaus müssen die gewonnenen Messdaten vom Anwender abgerufen werden können und schlussendlich soll die NPM-Maschine für Nano-Tooling-Anwendungen auch die Nutzung von Manipulatoren unterstützen.

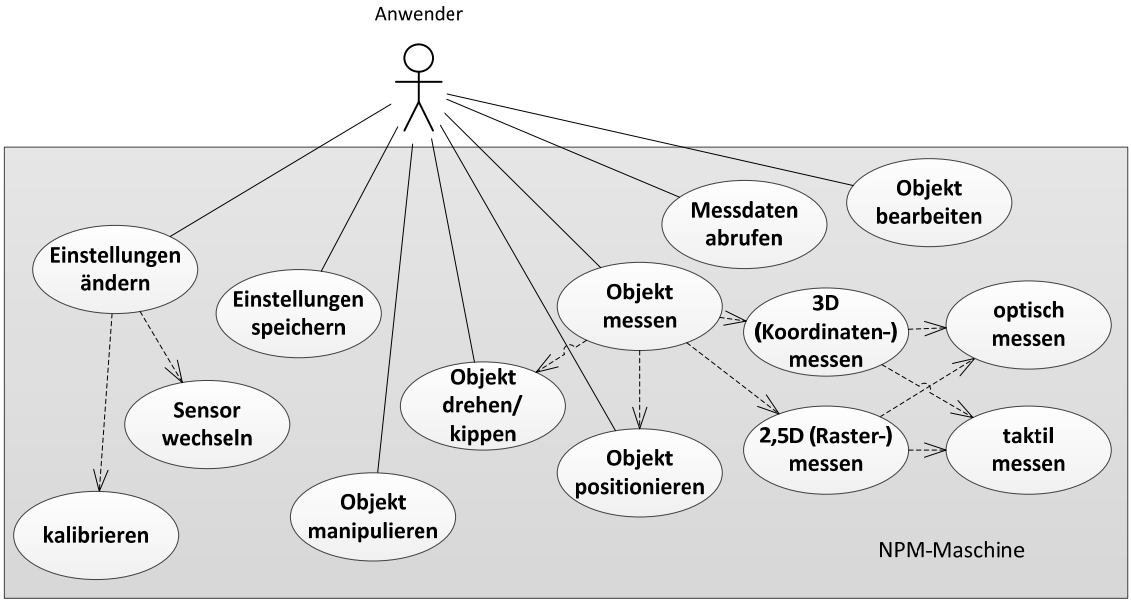


Abbildung 24: Anwendungsfälle für NPM-Maschinen

Aus den beschriebenen Nutzungsfällen lassen sich die folgenden grundlegenden Anforderungen ableiten:

Eine NPM-Maschine soll über ein mehrachsiges Positioniersystem, eine zusätzliche Rotationseinrichtung, taktile und optische Sensoren und eine Sensorwechseleinrichtung verfügen. Diese Anforderungen werden durch die iMERA noch weiter konkretisiert. Dementsprechend sind für die achsialen Stellbereiche Verfahrwege von mehreren 100 mm zu gewährleisten und die Präzision des Gesamtsystems soll mit verschiedenen Sensoren in den Sub-Nanometer-Bereich vordringen. Insbesondere die Anforderung, verschiedene Sensoren für die Lösung einer Messaufgabe anwenden zu können führt zur Entstehung weiterer Anforderungen an die Sensordatenverarbeitung und die Sensorreferenzierung. So ist es für die Fusion von Daten verschiedener Sensoren erforderlich, dass die NPM-Maschine über geeignete Kalibriermethoden für alle Sensoren verfügt und eine Möglichkeit zur Positionsreferenzierung der verschiedenen taktilen und optischen Einzel- und Parallel-Sensoren -

entweder im Rahmen der Kalibrierung oder nach einem Sensorwechsel - bereitstellt. Im Einzelnen können die Anforderungen an ein Steuerungskonzept für NPM-Maschinen wie folgt festgehalten werden:

- Einbinden unterschiedlicher mehrachsiger Positioniersysteme
- Einbinden zusätzlicher Kipp-/Rotationsachsen
  (z. B. für volle 3D-Messfähigkeit mit AFM-Sensoren)
- Bereitstellung multifunktionaler Nanoanalytik (taktile und optische Sensoren)
- Bereitstellung von Manipulatoren (Nanotools)
- Integration parallelmessender optischer Flächensensoren
- Anwendbarkeit des Gesamtsystems als Ultrapräzisions-KMG und 3D-AFM
- Austausch der Sensoren und Manipulatoren über einen automatisierten Wechsler
- Vollständige Parametrierbarkeit aller Komponenten und Berücksichtigung besonderer nanometrologischer Kalibrier- und Messdatenkorrekturaspekte
- Steuerung aller Hardwarekomponenten über eine gemeinsame generische Schnittstelle
- Vollständige Bewahrung des Funktionsumfangs der Hardware über die Schnittstelle hinweg
- Bereitstellen der Funktionalität für den Anwender über eine skalierbare Schnittstelle
- Anwenderfreundlichkeit durch klar strukturierte, moderne und intuitive Bedienkonzepte
- Steigerung der Sicherheit durch die Erkennung von Bedienfehlern

NPM-Maschinen sind damit im Grenzbereich zwischen Koordinatenmesstechnik und Rastersondenmikroskopie angesiedelt, integrieren optische Präzisionssensoren mit Nanometerauflösung als zusätzliche sensorielle Variante und ermöglichen darüber hinaus mit Nanotools die Bearbeitung von Objekten. Da weder koordinatenmesstechnische Steuerungskonzepte noch AFM-Ansätze alle Anforderungen einer multisensoriellen NPM-Maschine erfüllen, ist es unumgänglich, ein umfassendes, neues Konzept für die Ansteuerung einer iMERA konformen NPM-Maschine einzuführen. Eine erste einfache Übersicht zur Ausgestaltung eines derartigen Steuerungskonzept für NPM-Maschinen wird in Abbildung 25 gegeben.

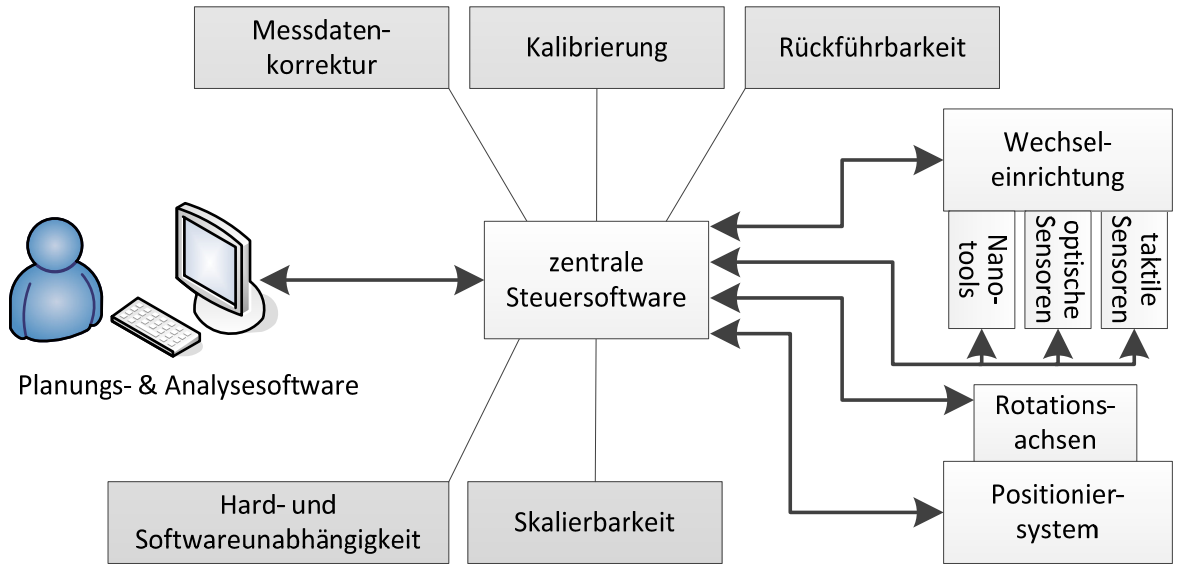


Abbildung 25: Anforderungen für das zu entwerfende Steuerungskonzept für NPM-Maschinen

Im Zentrum steht die zu entwerfende Steuerung für NPM-Maschinen. Diese muss über eine Reihe von Schnittstellen verfügen, um die vom Anwender gestellte komplexe Messaufgabe von der Messsoftware (links) in Form von Steuersequenzen korrekt an die metrologische Hardware (rechts) zu übermitteln. Neben dem Informationsfluss werden jedoch noch weitere Anforderungen - wie

Messdatenkorrektur, Kalibrierung der Sensoren, Systemskalierbarkeit und Hard- bzw. Softwareunabhängigkeit - an die Steuerung gestellt, welche zum Teil aus metrologischen Erfordernissen (dunkelgraue Rechtecke im oberen Teil von Abbildung 25) und architektonischen Notwendigkeiten (dunkelgraue Rechtecke im unteren Teil von Abbildung 25) erwachsen.

3.2 Modellierung des Steuerungskonzepts

Die Basis für Modellierung der NPMM-Steuerung bilden die erarbeiteten Anwendungsfälle einer NPM-Maschine. Da es sich bei NPM-Maschinen um hoch komplexe Systeme mit a priori nicht bekannter Hardware mit jeweils unterschiedlichen Eigenschaften und Anforderungen handelt, ist es zwingend erforderlich, die reale Hardware und ihre systemspezifische Ansteuerung vor dem Anwender zu verbergen. Wie in Abbildung 26 dargestellt ist, wird dafür die zentrale Steuerungssoftware als Hardwareabstraktionsebene eingeführt. Diese stellt dem Anwender eine klar definierte und funktional scharf umrissene Schnittstelle für die Steuerung von NPM-Maschinen bereit. So wird die tatsächliche Hardware des Systems für den Anwender als generische NPM-Maschine abstrahiert und es ist dem Nutzer möglich, das Messgerät ohne jedes Wissen über dessen Aufbau zu steuern. Das für die Hardwaresteuerung erforderliche Spezialwissen wird stattdessen als Softwarekomponente in der zentralen Steuerungssoftware hinterlegt. Die zentrale Steuerungssoftware stellt damit eine klassische Middleware dar, welche alle über die Anwenderschnittstelle empfangenen Steuerungsbefehle verarbeitet, diese in die hardwarespezifische Steuersequenzen umsetzt und anschließend über die gegebenenfalls proprietären Hardwareschnittstellen an die metrologische Hardware weiterleitet.

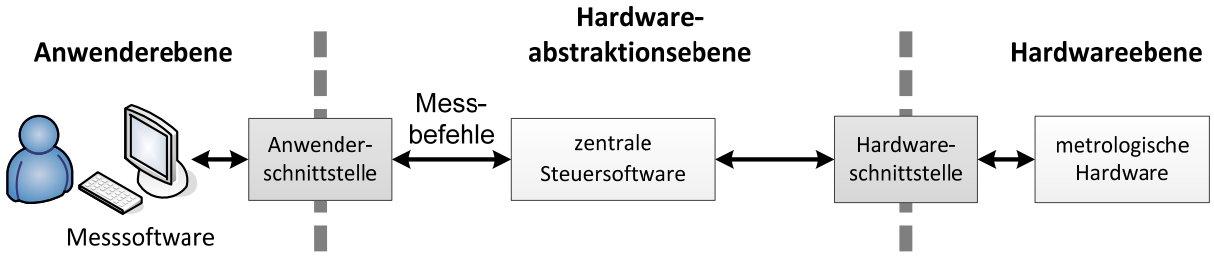


Abbildung 26: Mehrstufige Abstraktion der metrologischen Hardware durch die zentrale Steuersoftware aus Anwendersicht

Um dem Anspruch der Nanomesstechnik an höchste Präzision zu genügen, muss die korrekte Verarbeitung der gewonnenen Messdaten auf physikalische Größen stets sichergestellt sein, so dass erfasste Daten von der Datenverarbeitung der Hardwareabstraktionsschicht nicht beeinflusst werden. Aus dieser Forderung erwächst als weiterer Teilaspekt, dass alle verwendeten Sensorsysteme über die Anwenderschnittstelle kalibriert werden können, um systematische Messfehler korrigieren zu können. Als weiterer Aspekt erwächst daraus die Notwendigkeit, einmalig gewonnene Kalibrierdaten individuell und persistent für jeden Sensor zu speichern, um diese nach einem Neustart der NPM-Maschine erneut verwenden zu können und die oft langwierige Kalibrierprozedur nicht erneut ausführen zu müssen.

Die bisher getroffenen Festlegungen bezüglich der Modellierung der NPMM-Steuerung betrachten die NPM-Maschine als Blackbox ohne konkret spezifizierte metrologische Funktionalität. Je nach Realisierung des Aufbaus der konkreten NPM-Maschine stehen dem Anwender verschiedene Aktoren und Sensoren zur Interaktion mit einem Messobjekt zur Verfügung. Der abstrakte funktionale Unterschied verschiedener Realisierungen von NPM-Maschinen muss auch durch das Steuerungskonzept abbildbar sein. Das Ziel der weiteren Verfeinerung des Entwurfs ist somit die Schaffung einer maximalen Skalierbarkeit des Gesamtsystems NPM-Maschine bei gleichzeitigem

Erhalt der vollen Funktionalität der metrologischen Hardware und der Anwenderschnittstelle. Ein detaillierter Aufbau des Steuerungskonzepts für NPM-Maschinen ist in Abbildung 27 dargestellt und wird im Folgenden erläutert.

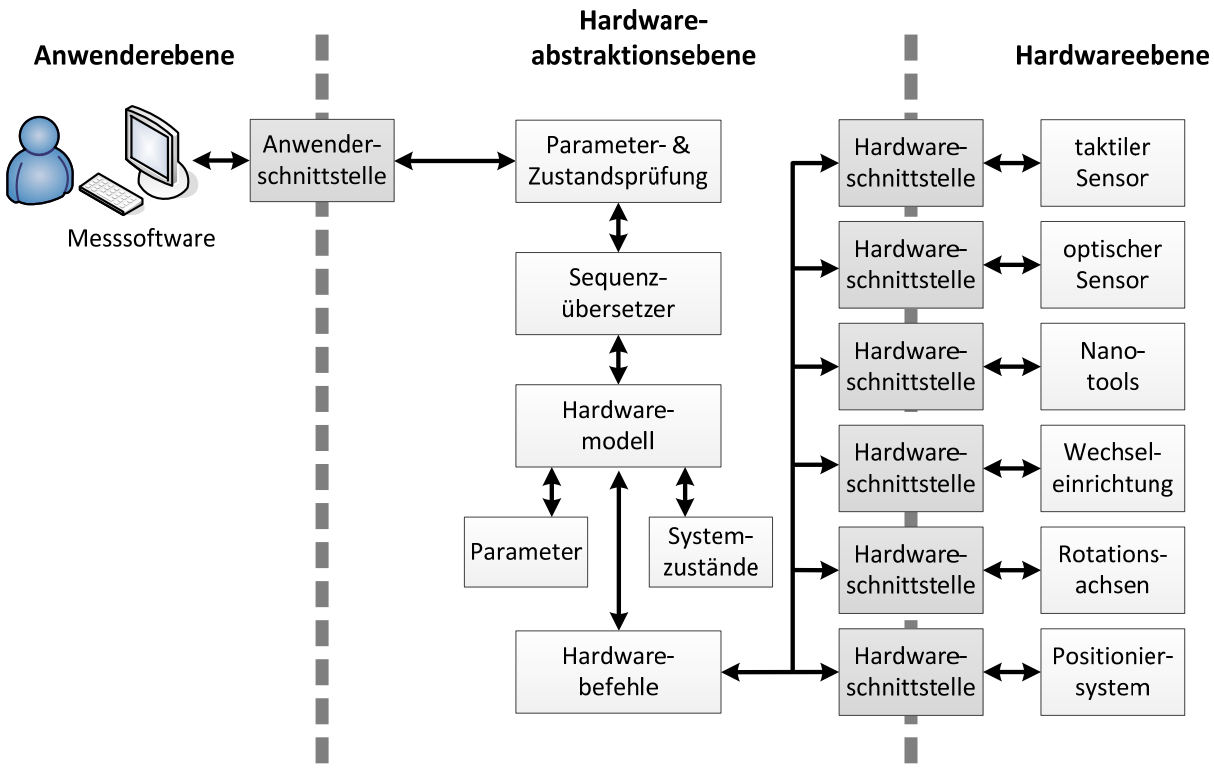


Abbildung 27: Aufbau der zentralen Steuerungssoftware als Hardwareabstraktionsebene

Um die maximale Skalierbarkeit des Steuerungskonzept zu gewährleisten, ist es zweckmäßig, die Integration der metrologischen Hardware zu modularisieren. Dazu wird das Hardwaregesamtsystem in Teilsystemgruppen untergliedert, um so Systemkomponenten mit vergleichbarer Funktionalität gruppenweise zusammenzufassen. Dieses Vorgehen vereinfacht in hohem Maße die Skalierbarkeit und vereinfacht die spätere Übertragung des Entwurfs in eine Steuerungssoftware. Durch die klare Abgrenzung von Eigenschaften und Funktionen der Teilsysteme können die komplexen metrologischen Komponenten vereinfacht als abstraktes Modul dargestellt und problemlos in die Steuerung aufgenommen werden. Das Modul stellt damit im Gesamtentwurf einen Platzhalter für ein konkretes System seiner Hardwaregruppen im Steuerungskonzept dar. Um die Platzhalterfunktion allgemeingültig für alle Systeme einer Hardwaregruppe zu gestalten, werden als weitere Schnittstellenebene die Hardwareschnittstellen in das Konzept aufgenommen. Ebenso wie die Anwenderschnittelle die konkrete NPM-Maschine abstrahiert, verallgemeinert jede Hardwareschnittstelle eine konkretes Hardware-Teilsystem auf einen definierten Funktionsumfang. Dadurch können alle Teilsysteme einer NPM-Maschine ohne Kenntnis des konkreten Aufbaus modellhaft beschrieben und in der NPMM-Steuerung abgebildet werden. Das so entstehende modulare Hardwaremodell ist für die NPMM-Steuerung von zentraler Bedeutung. Es bildet eine klar gegliederte hierarchische Darstellung der verfügbaren Teilsysteme jenseits der Hardwareschnittstelle ab. Gleichzeitig repräsentiert das Hardwaremodell sämtliche Parameter, Stellgrößen und Zustände sowie alle Interaktionsmöglichkeiten mit den Sensoren und Aktoren der konkreten NPM-Maschine und erlaubt gleichzeitig die maximale Skalierbarkeit des Gesamtsystems.

Die metrologische Funktionalität der NPM-Maschine wird dem Nutzer über die Anwenderschnittstelle bereitgestellt. Diese umfasst unter anderem Positionier-, Mess- und Parametrierfunktionen, welche in

Form von Schnittstellenbefehlen vom Anwender verwendet werden, um mit der metrologischen Hardware in Interaktion zu treten. Die Art und Weise der Interaktion von Anwender und NPMM-Steuerung wird von einem hardwareunabhängigen Schnittstellenprotokoll festgelegt, welches aus Sicht des Steuerungskonzeptes nicht weiter konkretisiert wird. Die konkreten Untersuchungen zur Umsetzung und Realisierung eines Protokolls für die Anwenderschnittstelle folgt in Kapitel 4.2.

Die hardwareunabhängige Interaktion über die Anwenderschnittstelle impliziert, dass vom Anwender versendete Befehlsaufrufe nicht unmittelbar für die Ansteuerung der NPMM-Hardware zur Anwendung kommen können. Einerseits sind die Befehle der Anwenderschnittstelle nicht direkt auf Befehle zur Steuerung der Hardware abbildbar. Vielmehr ist es erforderlich, die Schnittstellenbefehle in hardwareverständliche Befehlssequenzen zu übertragen und diese über die entsprechende Hardwareschnittstelle weiterzuleiten. Andererseits ist es wünschenswert, offensichtliche Bedienfehler durch eine vorgelagerte Parameter- und Zustandsprüfung fehlerhafter Aufrufparameter zu identifizieren und eine Abarbeitung von Befehlen nur dann zu ermöglichen, wenn der gegenwärtige Zustand der NPM-Maschine dies erlaubt. Für beide Aufgaben ist die modellhafte Abbildung der Hardwaresysteme von besonderer Bedeutung, da sie alle Zustände und Parameter modulweise verwaltet sowie in den Modulen auch die Steuerungsfunktionen für die Hardware bereitstellt.

Im weiteren Verlauf der Arbeit wird das in diesem Kapitel entworfene Steuerungskonzept für NPM-Maschinen in eine Steuerungssoftware für Nanopositionier- und Nanomessmaschinen überführt. Da das Steuerungskonzept nur allgemeine Festlegungen trifft, müssen die einzelnen Systemmodule der Hardwareabstraktionsebene sowie die daran angebundenen Schnittstellen konkretisiert werden. In den Folgenden Kapiteln werden die erforderlichen Untersuchungen zur Festlegung eines geeigneten Protokolls für die Anwender- und Hardwareschnittstelle dargelegt und es wird die hier festgelegte Struktur der Steuerungssoftware zu einer exemplarischen Realisierung eines nanometrologischen Koordinatenmessgerätes auf Basis von Nanomess- und Nanopositioniermaschinen weiterentwickelt.

Kapitel 4 Realisierung des Steuerungskonzepts

Das im vorigen Kapitel entworfene generische Steuerungskonzept für NPM-Maschinen wird im weiteren Verlauf dieses Kapitels konkretisiert und mit der Zielstellung, einen Demonstrator auf Basis von NPM-Maschinen aufzubauen, weiterentwickelt. Mit dem umgesetzten Demonstrator ist beabsichtigt, im weiteren Verlauf der Arbeit die Validierung und Leistungsbeurteilung des zugrundeliegenden Steuerungskonzepts im Einsatz realistischer metrologischer Applikationen zu beurteilen. Hierfür ist es erforderlich, die bewusst abstrakten Schnittstellen und Funktionsblöcke des Steuerungskonzepts zu konkretisieren, einen Entwurf für den Demonstrator zu entwickeln, diesen anschließend umzusetzen und als Gesamtsystem zu integrieren. Dementsprechend werden in den folgenden Abschnitten für alle Komponenten des Steuerungskonzepts geeignete Möglichkeiten zur Realisierung definiert. Soweit möglich wird dabei auf geeignete, bereits existierende Konzepte, Schnittstellen und Systeme zurückgegriffen und diese gegebenenfalls für die Aufgabenstellung der Nanometrologie adaptiert.

4.1 Konzeption des Demonstrators

In Abbildung 28 ist der Aufbau des Demonstrators veranschaulicht. Die Struktur des Demonstrators folgt dem im vorigen Kapitel entworfenen generischen Steuerungskonzept für NPM-Maschinen, wobei die einzelnen Komponenten weiter konkretisiert wurden.

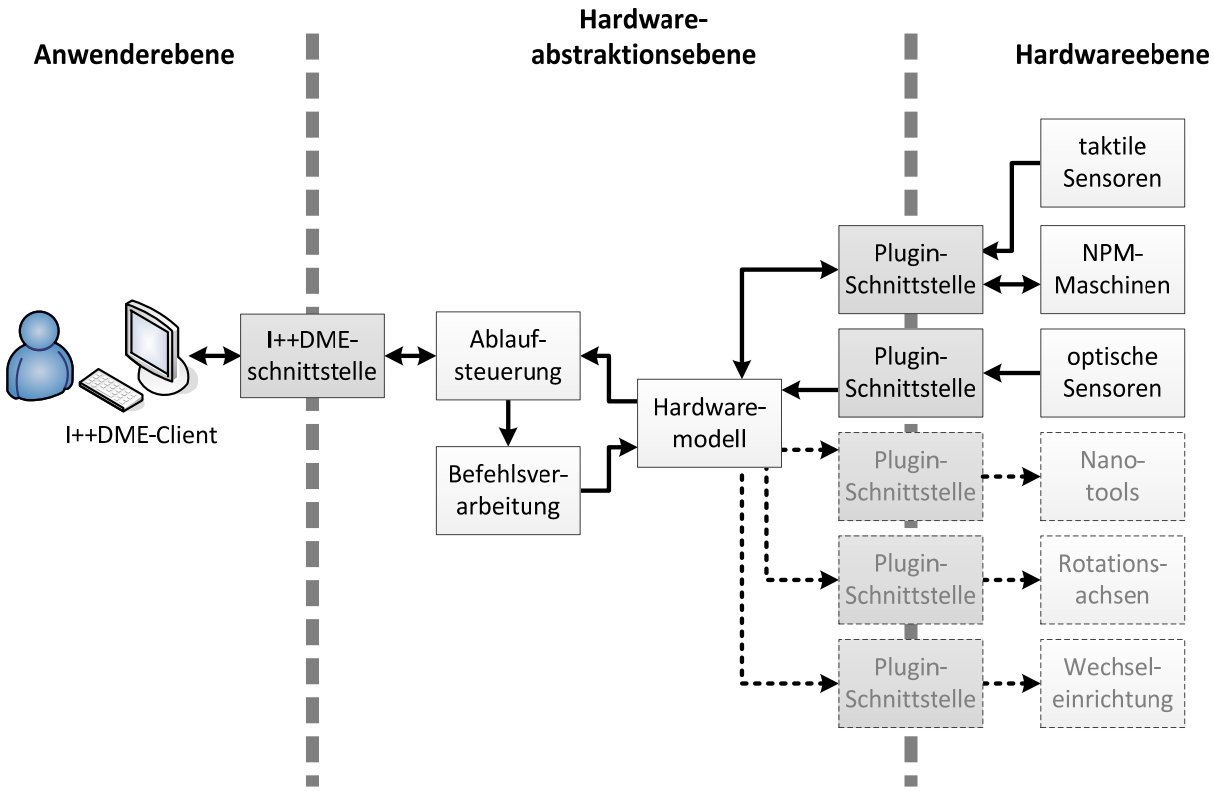


Abbildung 28: Konzept zur exemplarischen Umsetzung einer NPMM-Steuerung

Als Basis für die Anwenderschnittstelle soll das Mess- und Steuerungsprotokoll I++DME verwendet werden. Da I++DME dem Kontext der makroskopischen KMT entstammt, muss seine Eignung für die Nanomesstechnik zunächst validiert werden (siehe Abschnitt 4.2). Nanometrologische Aspekte, welche durch das KMT-Protokoll nicht hinreichend abgedeckt werden, müssen gegebenenfalls durch geeignete Erweiterungen der Schnittstellenspezifikation komplettiert werden.

Die Befehlsverarbeitung und Steuerung der Hardware obliegt der Hardwareabstraktionsebene, welche durch die Steuerungssoftware realisiert wird. Die Steuerungssoftware umfasst neben den Komponenten des generischen Steuerungskonzepts weitere Bestandteile, welche für den Empfang und die Verarbeitung von Schnittstellenbefehlen, Zuständen der Softwarekomponenten und der zu steuernden Hardware erforderlich sind. Weitere Ausführungen dazu sind im Abschnitt 4.3 zu finden.

Die Hardwareschnittstellen werden als dynamische Pluginschnittstellen realisiert. Der Mechanismus der Schnittstellen soll auf dem Einbinden von Dynamic Link Libraries (kurz DLLs) basieren. Dazu wird für jede Schnittstelle ein fester Funktionssatz zur Steuerung der Hardwarekomponente festgelegt, welcher von den Plugin-DLLs umgesetzt wird. Jedes Plugin umfasst seinerseits Steuerungssequenzen, welche die Funktionen seiner realisierten Plugin-Schnittstelle entsprechend den Anforderungen der zu steuernden Hardware umsetzen. Da DLLs während der Laufzeit zusätzlich zum Programmcode der zentralen Steuerungssoftware geladen werden können, wird dadurch eine einfache Möglichkeit zum schnellen und vollständigen Austausch der hardwarespezifischen Steuerungssequenzen im Falle eines Hardwarewechsels ermöglicht, ohne einen Eingriff in den zentralen Programmcode der Steuerungssoftware durchführen zu müssen. Insbesondere für das Messen mit optischen Sensoren ist das Pluginkonzept der Hardwareschnittstelle von besonderem Vorteil, da aufgrund des Multisensorkonzeptes der vollständige Austausch der gesamten sensorbezogenen Datenverarbeitung zur Laufzeit vollzogen wird.

Hardwareseitig soll der Demonstrator in der Lage sein, verschiedene NPM-Maschinen als Positioniersystem (sowohl der Typen NMM-1, als auch NPMM-200) zu steuern und taktile sowie optische Sensoren für metrologische Analysen zu unterstützen. Da NPM-Maschinen taktile Sensoren hard- und softwareseitig direkt einbinden, ihre Auslenkungssignale erfassen und verarbeiten sowie die gewonnenen Daten über die Steuerungsschnittstelle der NPM-Maschine übertragen können, entfällt bei der Implementierung des Demonstrators die Umsetzung einer gesonderte Hardwareschnittstelle für taktile Sensoren. Die Entwicklungsarbeiten der übrigen metrologischen Komponenten für NPM-Maschinen (Nanotools, Sensorwechsler und Rotationsachsen) sind zum gegenwärtigen Zeitpunkt noch nicht hinreichend fortgeschritten, um ihre Steuerung über eine dezidierte Automatisierungsschnittstelle bereitzustellen. Diese Komponenten werden im Demonstrator daher nicht vollständig umgesetzt.

Da das Steuerungskonzept die Funktionalität dieser Komponenten jedoch bereits abbildet, ist es ohne weiteres möglich, diese Komponenten zu einem späteren Zeitpunkt zu realisieren. Im Rahmen dieser Arbeit wird dafür in Abschnitt 4.4.3 ein Integrationskonzept beschrieben, wie neue bzw. zum gegenwärtigen Zeitpunkt nicht berücksichtigte Komponenten in die NPMM-Steuerung aufgenommen werden können.

Der Aufbau der folgenden Darlegungen lehnt sich an die Grafik in Abbildung 28 an. Das Konzept wird vom Anwender ausgehend ebenenweise verfeinert, wobei wegen der großen Komplexität ein besonderes Augenmerk auf die Komponenten der Hardwareebene gelegt wird.

4.2 I++DME als Anwenderschnittstelle

Im Jahr 2009 zeigte Dai [71], dass es prinzipiell möglich ist, eine NPM-Maschine mit einer I++DME fähigen Messsoftware für nanometrologische Anwendungen zu steuern. Es gelang ihm, die Befehlssätze von I++DME und der NPM-Maschine über einen Interpreter soweit ineinander zu überführen, dass die Maschine positioniert und einzelne Punkte angetastet werden konnten. Dieser Ansatz soll im Rahmen dieser Arbeit aufgegriffen und weiterentwickelt werden. Da Scanmessungen ein integraler Bestandteil nanometrologischer Analysen sind, ist es zwingend erforderlich, diesen Aspekt in die Steuerungsschnittstelle von NPM-Maschinen zu integrieren.

4.2.1 Befehlsumfang von I++DME

I++DME bietet einen umfassenden und sehr breit gefächerten Befehlssatz für den Einsatz in der makroskopischen KMT. Von zentraler Bedeutung sind die Basisbefehle für Bewegung, Punkt-, und Scanmessungen. Sie ermöglichen das freie Positionieren von Messobjekt und Sensor sowie die Antastung von Einzelpunkten oder Punkten entlang einer zu scannenden geometrischen Primitiven - ein Kreis oder eine Linie. Im Fall von Scans an Freiformflächen erfolgt die Bewegung in einer Ebene oder auf einem Zylindermantel. Optional wird die Möglichkeit zur Steuerung eines einachsigen Drehtisches unterstützt. Seit der 2009 veröffentlichten Version 1.7 unterstützt I++DME das Multisensorkonzept und die softwaregestützte Sensorverwaltung. I++DME-konforme Steuerungssoftware stellt damit eine flexible, vom Anwender editierbare Datenbank für die Verwaltung beliebiger taktiler Sensorsysteme bereit. Der Anwender hat die Möglichkeit, in einer Datenbank hinterlegte Sensoren einzeln zu parametrieren, für Messaufgaben zu selektieren und damit eine Antastung durchzuführen. Ferner ermöglicht I++DME die Definition von Koordinatentransformationen, wodurch das Positionieren des KMGs vereinfacht wird. Durch die Festlegung einer Koordinatentransformation wird das intrinsische Koordinatensystem des KMGs verschoben, wodurch es an die Messaufgabe angepasst werden kann. Zuletzt stellt I++DME noch Befehle zum Festlegen von verbotenen Bereichen sowie zur Kompensation thermisch bedingter

Ausdehnung bereit und unterstützt die ereignisorientierte Meldung von Mess- und Positionsdaten an die Nutzersoftware.

Da I++DME eine Entwicklung aus dem Umfeld der makroskopischen Koordinatenmesstechnik ist, sind die Steuerungsaspekte, respektive der Befehlsumfang, aus Sicht der nanometrologischen KMT erfüllt. Im Hinblick auf AFM-Messungen unterstützt I++DME jedoch keine Befehle, welche entsprechende Mess- bzw. Positionierregime im Ganzen bereitstellen, um eine Oberfläche ad-hoc mit einem 2,5D-Rasterscan zu untersuchen. Für AFM-Scans werden üblicherweise Kamm- oder Meandertrajektorien gefahren und während der Bewegung fortlaufend äquidistant Messdaten erhoben. Auch wenn I++DME keinen Einzelbefehl für diese Messaufgabe unterstützt, kann eine geeignete Trajektorie mit rasterartig verteilten Messpunkten durch die Kombination von Mess- und Positionierbefehlen aus dem I++DME Befehlssatz synthetisiert werden. Wie in Abbildung 29 dargestellt, können die Trajektorien durch die Kombination von alternierenden Linienscans (hell dargestellt) und Positionierbewegungen (dunkel dargestellt) generiert werden.

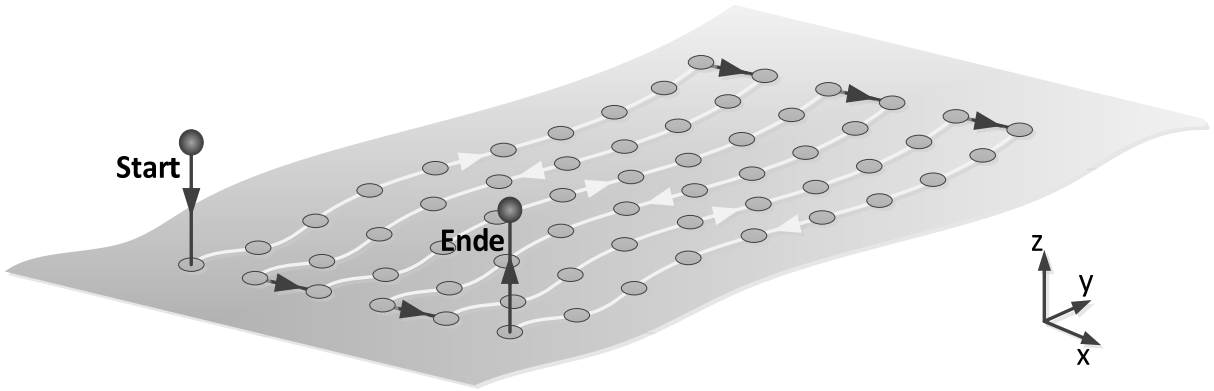


Abbildung 29: Kombination mehrerer Linienscans zu einer Meandertrajektorie

4.2.2 Darstellung numerischer Daten in Anwenderschnittstelle und Steuerungssoftware

Das I++DME Protokoll legt weder explizit noch intrinsisch eine Auflösungsgrenze für die zu übertragenden Messdaten und Befehlsparameter fest. Positions- und Längendaten werden mit beliebiger Genauigkeit als lesbare Werte einer textbasierten Befehls- bzw. Messdatensequenz zwischen Client und Server übermittelt. Als einzige Festlegung bezüglich der Darstellung von Positions- und Längendaten wird durch I++DME das Millimeter als Basisgröße definiert. Faktisch ist das I++DME Protokoll damit bezüglich seines Auflösungsvermögens geeignet, um als Anwenderschnittstelle für nanometrologische Applikationen Anwendung zu finden.

4.2.3 Erweiterung des Befehlsumfangs der Anwenderschnittstelle

Der I++DME Befehlssatz der Version 1.7 ist der Standardbefehlssatz der Steuerungssoftware. Er bildet die gemeinsame Schnittmenge zur Steuerung von NPM-Maschinen mit kommerzieller I++DME konformer Software. Der I++DME-Funktionsumfang ist hinreichend, um die Hardware einer NPM-Maschine zu parametrieren und beliebige nanometrologische Untersuchungen mit taktilen 3D- und 2,5D-messenden Rastersensoren zu ermöglichen.

Dennoch sind Erweiterungen des I++DME-Basisbefehlssatzes erforderlich, um den uneingeschränkten Zugriff auf das Positioniersystem NPM-Maschinen zu gewährleisten. Der Befehlssatz von NPM-Maschinen besitzt zwar mit I++DME eine große Schnittmenge, ist den Anforderungen der Nanometrologie und Parametrierbarkeit des Positioniersystems den einfachen Möglichkeiten von

I++DME überlegen. Insbesondere im Hinblick auf die Weiterverwendung der zahlreich existierenden Matlab-Steuerungsskripte für NPM-Maschinen, welche auf deren proprietären Befehlssatz aufbauen, ist die Integration des NPM-Maschinen-Befehlssatzes in die Anwenderschnittstelle erforderlich.

Problematisch ist allerdings die fehlende I++DME-Unterstützung für optische Sensoren, da diese Sensorklasse für metrologische Analysen im makroskopischen Bereich noch keine hinreichende Verbreitung gefunden hat. In Verbindung mit der Erweiterung *Optical Sensor Interface Standard* (kurz: OSIS) können mit optischen Sensorsystemen zwar punktbasierte Einzelpunkt- oder Scanmessungen über I++DME ausgeführt werden, jedoch ist es damit nicht möglich, die Vorteile optischer Sensoren vollständig auszuschöpfen. Speziell für die Klasse der optisch parallel messenden Flächen- und Liniensensoren müssen neue Steuerbefehle entworfen werden. Durch diese Erweiterung soll es neben der Erweiterung über OSIS ermöglicht werden, die volle Funktionalität optischer Sensoren für metrologische Anwendungen zu nutzen.

4.3 Aufbau der Hardwareabstraktionsebene

Im Sinne von I++DME sind Befehle über ein Netzwerk übertragene Zeichenketten, welche von der Schnittstellenimplementierung empfangen, interpretiert, analysiert und verarbeitet werden müssen. Die Ausführung der einzelnen Schritte übernehmen Datenverarbeitungskomponenten, welche den bisherigen generischen Entwurf der Hardwareabstraktionsebene konkretisieren (siehe Abbildung 30).

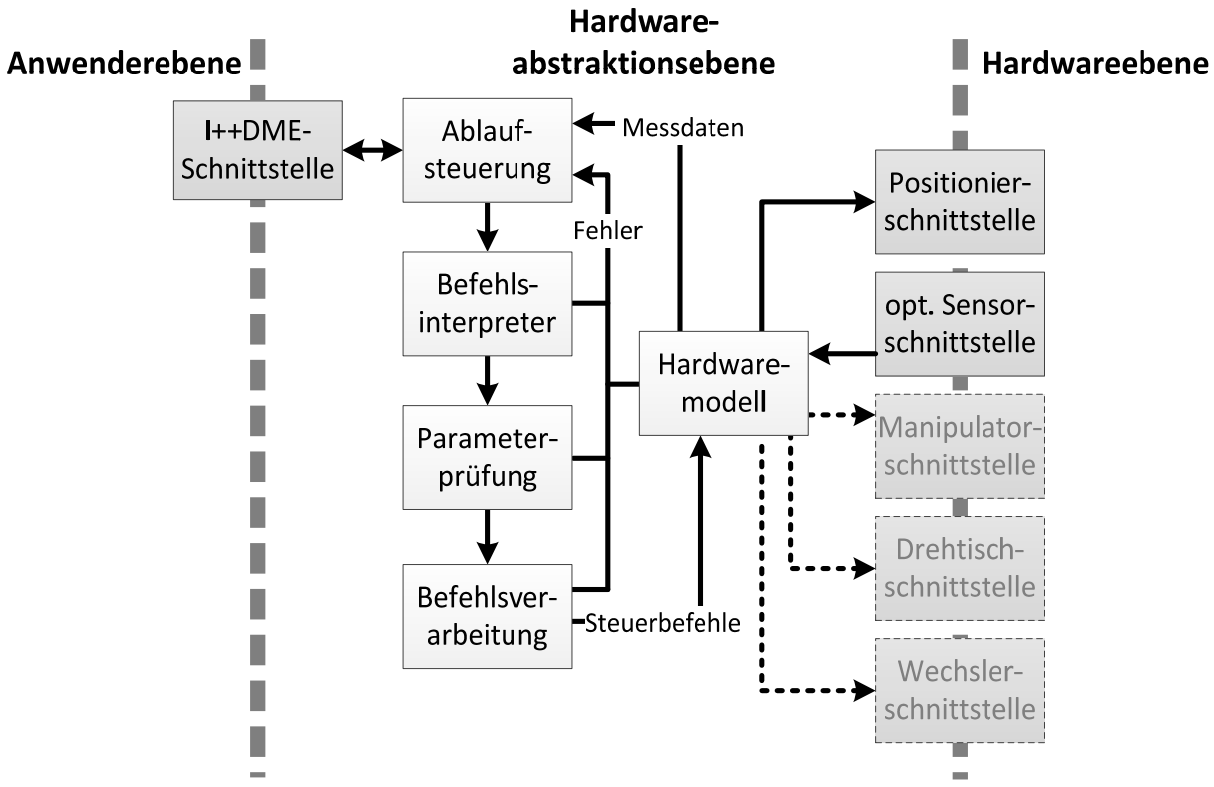


Abbildung 30: Datenfluss in der Hardwareabstraktionsebene der NPMM-Steuerung

Für die Interaktion mit dem Anwender wird die Ablaufsteuerung als Komponente eingeführt. Mit ihr baut die Steuerungssoftware eine TCP-Netzwerkverbindung mit dem Anwender auf und ermöglicht so den Empfang von Befehlen und das Senden von Daten. Alle von der Ablaufsteuerung empfangenen Aufrufe werden dem Stringinterpreter übergeben. Dieser analysiert die Zeichenkette auf syntaktische Korrektheit, extrahiert bei erfolgreicher Prüfung den auszuführenden Befehl und übergibt die Aufrufparameter an die Parameterprüfung. Durch Abfragen des Hardwaremodells ermittelt die

Parameterüberprüfung die logische Korrektheit der übergebenen Daten und prüft, ob der Maschinenzustand die Abarbeitung des Befehls erlaubt. Verlaufen alle Prüfungen positiv, kommt der Befehl in der Befehlsverarbeitung zur Ausführung. Werden für die Ausführung zusätzliche Parameter (z. B. für die Bahngenerierung) benötigt, werden diese aus dem Datenbestand des Hardwaremodells abgerufen.

Die Steuerung der NPM-Maschine erfolgt durch die Befehlsverarbeitung über eine transparente Hardwareschnittstelle. Um verschiedene Generationen der NPM-Maschine steuern zu können, wird eine Maschinenschnittstelle implementiert, welche die NPM-Maschinenbefehle über das entsprechende Kommunikationsmedium an die NPM-Maschine übergibt. Eine weitere Schnittstellenkomponente ist die optische Sensorschnittstelle, über welche beliebige optische Sensoren transparent in die Steuerungssoftware eingebunden werden können. Weitere vorgesehene Schnittstellen dienen der Ansteuerung von Sensorwechsler, Drehtischen und Nano-Manipulatoren. Da die strukturelle Zuordnung dieser Komponenten im Architekturmodell auch im zentralen Hardwaremodell zu suchen ist, werden die Schnittstellen ebenfalls an diese Komponente gekoppelt.

4.3.1 Architektur der Steuerungssoftware

In Abbildung 31 ist die architektonische Struktur der zentralen Steuerungssoftware des Demonstrators dargestellt. Die Komponenten des Steuerungskonzepts werden durch Klassen weiter untergliedert, wodurch die komplexen Verarbeitungsalgorithmen gekapselt und übersichtlich strukturiert werden können. Aufgrund ihrer komplexen und umfangreichen Funktionalität werden die Ablaufsteuerung und das Hardwaremodell aus einer mehrschichtigen Architektur zusammengesetzt. Die zentrale Klasse des Hardwaremodells bildet das KMG-Modell. Dieses umfasst die softwaretechnische Abbildung der Parameter des kartesischen Positioniersystems NPM-Maschine, eine Repräsentation der Messobjektparameter (Klasse Messobjekt) und die skalierbare Sensordatenbank (Klasse Wechsler), welche Kalibrierdaten, Sensorparameter und die Position des Sensors in der Wechselvorrichtung speichert.

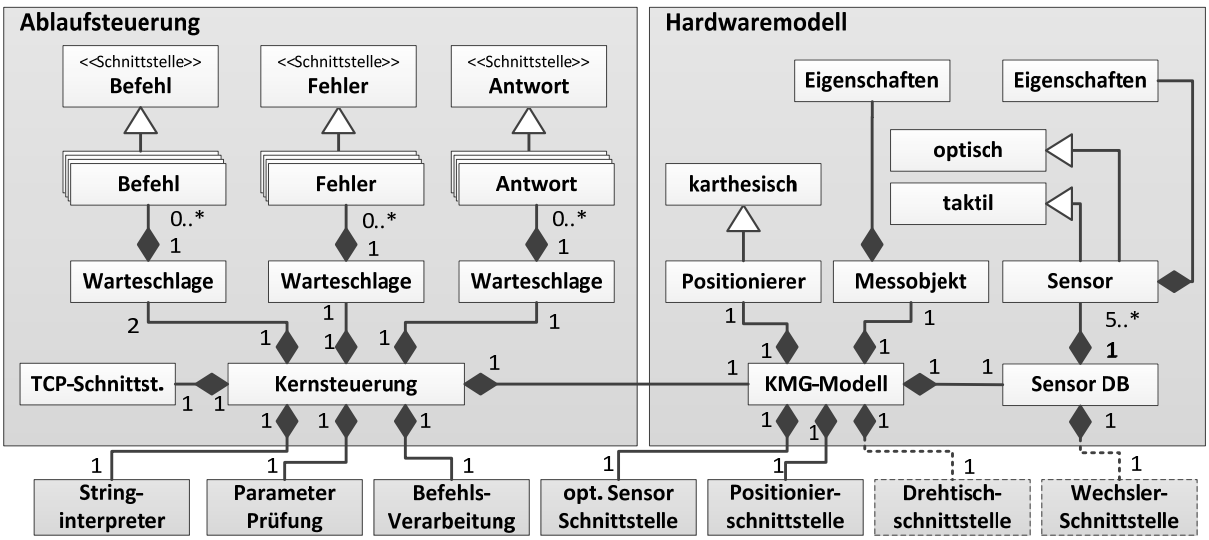


Abbildung 31: Klassendiagramm der zentralen Steuerungssoftware

Analog dazu wird die Komponente Ablaufsteuerung um die Kernsteuerung, welche die Datenübertragungsmechanismen bereitstellt und die Befehlsabarbeitung organisiert, aufgebaut. Im laufenden Betrieb bedient sich diese Klasse zweier Befehlswarteschlangen, um Anwenderaufrufe als Befehlsobjekt priorisiert zu puffern, bevor sie von der Befehlsverarbeitung übernommen werden. Darüber hinaus werden Messdaten und abgefragte Parameter als Antwortobjekte und interne bzw.

externe Störzustände als Fehlerobjekte repräsentiert, welche vor dem Versenden über die Anwenderschnittstelle in separaten Warteschlagen zwischengespeichert werden. Um die Handhabung mehrerer Befehls-, Fehler- und Antwortklassen zu ermöglichen, erbt jede Klassengruppe ihre Methodensignatur von einer gruppeneigenen Schnittstelle. Die spezifische Funktionalität der Methoden wird von jeder abgeleiteten Klasse gesondert implementiert. So ist sichergestellt, dass alle Objekte inhaltsunabhängig verarbeitet werden können.

Das Konzept für den sequentiellen Ablauf einer Befehlsverarbeitung im laufenden Betrieb ist in Abbildung 32 dargestellt. Vom Anwender asynchron versendete Aufrufe werden vom Stringinterpreter empfangen und zu einem Befehlsobjekt konvertiert. Dem Stringinterpreter obliegt dabei nur die Verifikation der syntaktischen Korrektheit des Aufrufs. Die logische Überprüfung des Befehls erfolgt anschließend unter Zuhilfenahme des Hardwaremodells durch die Parameterprüfung. Wurde der Aufruf erfolgreich zu einem Befehl konvertiert wird eine Empfangsquittierung als Antwort erzeugt und das Befehlsobjekt in die entsprechende Warteschlange eingereiht. Bei Misserfolg wird das Befehlsobjekt zerstört und ein Fehlerobjekt in der entsprechenden Warteschlange zum Versenden abgelegt. Da die Warteschlangen nur der Synchronisierung dienen und keinen Einfluss auf die Datenverarbeitung nehmen, sind sie in Abbildung 32 nicht dargestellt.

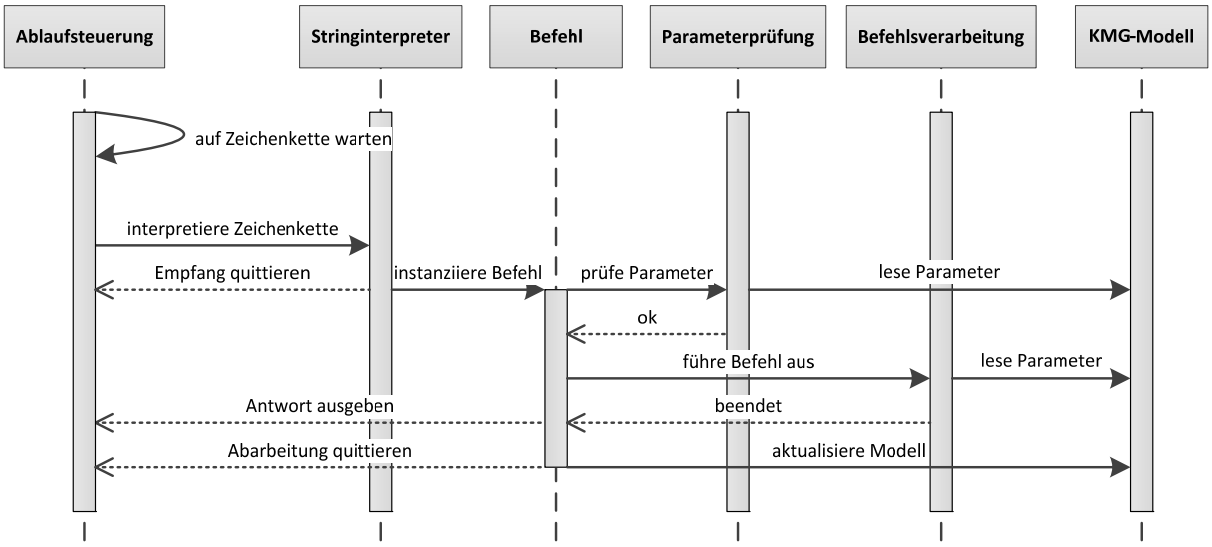


Abbildung 32: Interaktionssequenz bei der Befehlsabarbeitung in der Hardwareabstraktionsebene

Die weitere Verarbeitung im Erfolgsfall geschieht durch die Klasse Befehlsverarbeitung. Sie entnimmt der Warteschlange den höchst priorisierten bzw. den ersten Befehl und führt ihn aus. Im Rahmen der Befehlsabarbeitung werden die Aufrufparameter aus dem Befehlsobjekt ausgelesen und wenn notwendig durch zusätzliche Parameter aus dem Hardwaremodell ergänzt. Mess- oder Bewegungsbefehle treten zusätzlich über die Plugin-Schnittstellen mit der NPM-Maschine und den Sensoren in Interaktion. Abschließend wird geprüft, ob die Parameter des Hardwaremodells aktualisiert werden müssen und es wird ein weiteres Antwortobjekt in der Warteschlange abgelegt, um die erfolgreiche Abarbeitung des Befehls zu quittieren.

4.3.2 Numerische Auflösung der Steuerungssoftware

Im praktischen Einsatz können numerische Daten nicht mit unendlicher Genauigkeit verarbeitet werden. Die erreichbare numerische Auflösung eines Datenverarbeitungsalgorithmus wird durch die endliche Speicher- und Datenverarbeitungskapazität der verwendeten Hardware beschränkt. Die folgende Betrachtung zur erreichbaren numerischen Auflösung basiert auf der Annahme, dass gemessene Daten als genormte Gleitkommazahlen gemäß IEEE 754 [72] verarbeitet werden.

Festkommadatentypen sind wegen ihres starren Wertebereichs für metrologische Applikationen nicht geeignet. Bei Berechnungen von komplexen metrologischen Objektkennwerten können Überläufe auftreten, wodurch das berechnete Ergebnis verfälscht und damit unbrauchbar wird.

Die Norm IEEE 754 hingegen definiert Gleitkommadatentypen mit einfacher und doppelter Genauigkeit, welche den numerischen Wert in Exponentialdarstellung mit unterschiedlicher Datenbreite für Mantisse und Exponent wiedergeben (siehe Tabelle **11**). Aus der Darstellung der Daten im Speicher kann die relative Auflösung der Datentypen mit $\varepsilon_{rel} = 2^{-24} \approx 6 \cdot 10^{-8}$ für einfache Genauigkeit und $\varepsilon_{rel} = 2^{-53} \approx 1{,}1 \cdot 10^{-16}$ für die Darstellung mit doppelter Genauigkeit abgeschätzt werden.

Datentyp Genauigkeit	Mantisse [Bit]	Exponent [Bit]
einfach	23	8
doppelt	52	11

Tabelle 11: Darstellung der IEEE 754 Gleitkomma-Datentypen

Der Fließkommadatentyp einfacher Genauigkeit ermöglicht mit einer Auflösung von 60 nm keine hinreichende Granularität, um nanometrologische Messdaten abbilden zu können. Deshalb werden metrologischen Daten in der Steuerungssoftware mit dem IEEE 754 Gleitkommatyp doppelter Genauigkeit repräsentiert, welcher mit einer Auflösung von rund 0,1fm hinreichend ist.

4.4 Einbettung und Umsetzung der Hardwareschnittstellen

Der Zugriff der Steuerungssoftware auf die eigentliche Hardware der NPM-Maschine erfolgt über die verschiedenen Hardwareschnittstellen. Diese sind erforderlich, um die Datenverarbeitungs- und Steuerungslogik in der Hardwareabstraktionsebene losgelöst von den proprietären Anforderungen zur Steuerung konkreter Hardware entwickeln zu können. Durch diese Modularisierung und die Loslösung von konkreter Hardware kann die Software besser strukturiert, ihre Wiederverwendbarkeit verbessert und die Steuerung - nicht zuletzt durch die vielen Schnittstellen - sehr einfach skaliert werden.

Wiederverwendbarkeit und Skalierbarkeit des Gesamtsystems NPM-Maschine im Allgemeinen und des zentralen Softwaresystems im Speziellen gehen dabei Hand in Hand. Da die konkrete Umsetzung jeder proprietären Hardwaresteuerung in einen Schnittstellenadapter der Hardwareebene ausgelagert wird, kann die Hardware der NPM-Maschine zur Laufzeit ausgetauscht werden. Diese Anforderung ist bei einigen Komponenten, wie den Sensoren, als obligatorisch anzusehen. Beim Positioniersystem und den metrologischen Zusatzkomponenten ist dies jedoch nicht zwingend erforderlich. Allerdings ermöglicht die komponentenweise Substitution der Hardware die einfache Skalierung des Gesamtsystems, einerseits in Bezug auf seine Parameter (z. B. Länge der axialen Verstellwege) und andererseits aber auch im Hinblick auf den Aufbau des Systems an sich. So werden Komponenten, deren Schnittstellen nicht durch eine Hardwarekomponente abgedeckt werden von der zentralen Steuerung erkannt und ihre Funktionalität für den Anwender im laufenden Betrieb nicht verfügbar gemacht.

Im zu implementierenden Demonstrator werden wegen des hohen zeitlichen Implementierungsaufwands nur einige Schnittstellen des Steuerungskonzepts exemplarisch realisiert. Die Umsetzung von Schnittstellen beschränkt sich auf die Steuerung von Positioniersystemen sowie die Integration taktiler und optischer Sensoren, weshalb die damit verbundenen Schnittstellenadapter und die Optionen zu deren Integration in die Steuerungssoftware im Folgenden detailliert erläutert

werden. Da alle Schnittstellen dem zentralen Abstraktionskonzept folgen, kann die vorgestellte Vorgehensweise auch auf die nicht umgesetzten Schnittstellen angewandt werden. Diese Schnittstellen werden im folgenden jedoch hinreichend konzeptioniert. Sie repräsentieren damit Hardwarekomponenten, welche vom Demonstrator unterstützt werden, dem Anwender aber zur Verdeutlichung der Systemskalierung noch nicht zur Verfügung stehen.

4.4.1 Anbindung von NPM-Maschinen als Positioniersystem

Eine Anforderung an den Demonstrator ist die Fähigkeit, verschiedene Typen von NPM-Maschinen als Positioniersystem einbinden und nutzen zu können. Die beiden Generationen der NPM-Maschine benötigen jedoch unterschiedliche Treiber und Medien für die Datenübertragung. Damit alle NPM-Maschinen mit dem Demonstrator gesteuert werden können, wird die Positionierschnittstelle als Abstraktionsschicht für das Positioniersystem eingeführt und die eigentliche Interaktion mit der jeweiligen NPM-Maschine auf eine externe Kommunikationskomponente, das Positionier-Plugin ausgelagert. Das Positionierplugin für eine NPM-Maschine wird als Funktionsbibliothek [73] (engl. Dynamic Link Library, kurz DLL) mit dem Funktionsumfang der Positionierschnittstelle umgesetzt, wodurch der Programmcode für die Steuerung des Positioniersystems flexibilisiert wird und zur Laufzeit ausgetauscht werden kann. Dadurch wird der Entwurf modularisiert, ist über den Pluginmechanismus der Schnittselle skalierbar und damit offen für zukünftige Erweiterungen. Durch die Auslagerung der Kommunikationsmechanismen in Plugins, können perspektivisch weitere Positioniersysteme für den Demonstrator umgesetzt werden, wofür lediglich eine zusätzliche schnittstellenkonforme Plugin-DLL erstellt werden muss.

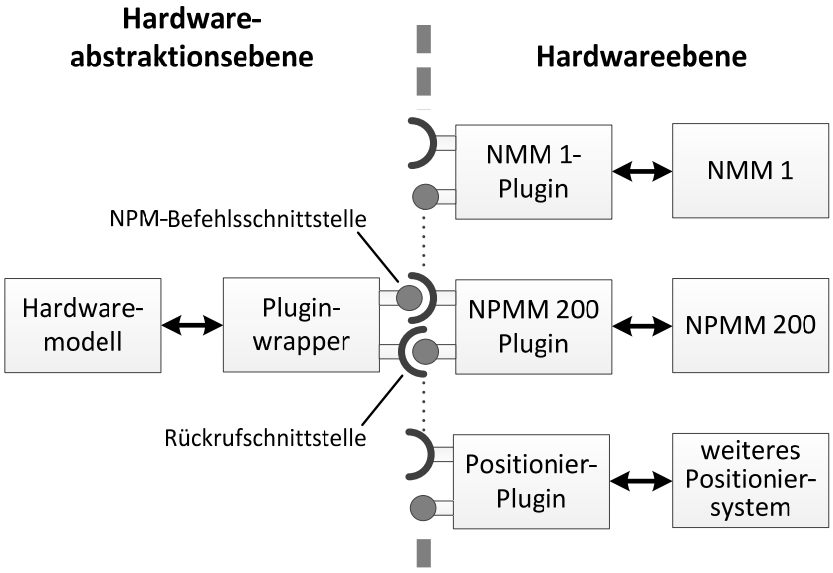


Abbildung 33: Positionierschnittstelle als Statthalter für austauschbare Kommunikations-Plugins

Die Integration der Positionierschnittstelle in die Steuerungssoftware erfolgt durch eine Wrapperklasse. Diese Klasse ist während der Programmausführung in der Lage, ein Plugin in den Speicherbereich der Steuerungssoftware zu laden und dessen Funktionalität für die Steuerungssoftware im internen Hardwaremodell bereitzustellen. Die Klasse *Pluginwrapper* stellt damit, wie im Ausschnitt des Demonstratorkonzepts in Abbildung 33 dargestellt, einen Statthalter für die tatsächliche NPM-Maschine dar und realisiert die Abstraktion der Positionierschnittstelle. Die Hardwareabstraktionsebene richtet Mess- und Positionieranfragen ausschließlich an den Statthalter, welcher sie seinerseits über die geladene Plugin-DLL an das Positioniersystem weiterleitet.

Das Suchen und Einbinden von kompatiblen Positionierplugins erfolgt vollautomatisiert durch die Statthalterklasse. Damit ein Plugin als solches identifiziert und von anderen DLLs unterschieden werden kann, besitzen alle Positionierplugins spezielle Identifikationsfunktionen, mit welchen sie sich als Positionieradapter ausweisen und der Steuerungssoftware Informationen für ihre Adressierung übergeben. Im Rahmen der automatisierten Pluginsuche wird versucht, alle gefundenen DLLs über die Identifikationsfunktion als Positionierplugin anzusprechen. Im Erfolgsfall werden die übergebenen Adressinformationen für den DLL-Ladeprozess der Wrapperklasse gespeichert.

Da NPM-Maschinen eine unmittelbare Unterstützung für das Einbinden taktiler Sensoren und der Verarbeitung der im Rahmen einer taktilen Messung anfallenden Daten bereitstellen, ist es zweckmäßig, die Hardwareschnittstelle für taktile Sensoren mit der Positionierschnittstelle zu vereinen. Für die ereignisgesteuerte Übertragung von Messdaten und Fehlerzuständen wird die Schnittstelle um mehrere Callback-Funktionen ergänzt. Diese Callbacks (engl. Rückrufe) stellen inverse Schnittstellen dar. Die Callback-Funktionen werden an die aufgerufene Software übergeben, um im weiteren Verlauf der Datenverarbeitung die Umkehr des Kontrollflusses zu ermöglichen. Die Kontrollflussinversion ist insbesondere bei der Übertragung asynchroner Daten günstig, da hierdurch der Zeitpunkt der Übertragung von der Datenquelle bestimmt wird, die Datensenke jedoch weiterhin die Art und Weise der Datenverarbeitung kontrollieren kann. Insbesondere für die Messtechnik ist dieses Vorgehen sinnvoll, da die Messdaten nicht von der Datenquelle gepuffert werden müssen, wodurch die Verarbeitungsalgorithmen übersichtlich gestaltet und der Datendurchsatz optimiert werden können.

4.4.1.1 Auflösung von I++DME-Beschränkungen

Da der I++DME-Befehlssatz und der Funktionsumfang der Maschinenschnittstelle nicht identisch sind, müssen einige Konventionen festgelegt werden, um I++DME Befehle konsistent in die umfangreichen parametrierbaren Aufrufe der Positionierschnittstelle überführen zu können. Dazu müssen von I++DME nicht abgebildete Parameter auf gültige Standardwerte festgelegt werden. Die Notwendigkeit für dieses Vorgehen ist bei allen Messbefehlen erforderlich. Positionier- und Parametrierbefehle umfassen in der Regel identische Aufrufparameter bzw. sind durch I++DME nicht abgebildete Parameter für die Steuerung des Positioniersystems unerheblich. Im Folgenden werden die Parameter von Messbefehlen betrachtet. Sowohl I++DME als auch der Befehlssatz von NPM-Maschinen untergliedern eine Messung in vier Phasen:

1. Das Annähern an die Oberfläche (englisch: Approach)
2. Das Suchen der Oberfläche (englisch: Search)
3. Die eigentliche Messung am Objekt (englisch: Measurement)
4. Das Lösen vom Objekt (englisch: Retract)

Das I++DME Protokoll definiert in allen Phasen einen eigenen Entfernungsparameter sowie eine einheitliche Maximalgeschwindigkeit und -beschleunigung für die Phasen 1,2 und 4 bzw. einen separaten Parametersatz für die eigentliche Messung. Im Gegensatz dazu erfordern die NPM-Befehle zusätzlich einen Parametersatz, welcher den maximal erlaubten Ruck für jede der vier Messphasen festlegt. Um das Parameterdefizit beim Umsetzten der I++DME-Befehle in die Schnittstellenbefehle zu überwinden, werden die Maximalgeschwindigkeit und die maximal erlaubte Beschleunigung in den Phasen 1 und 2 auf die I++DME Parameter festgelegt. Um die Wartezeit beim Entfernen des Tasters vom Objekt zu verkürzen, wird die Retract-Geschwindigkeit auf den dreifachen Wert erhöht.

$$v_{Approach} = v_{Search} = \frac{1}{3} v_{Retract} = v_{I++DME} \tag{4.1}$$

$$a_{Approach} = a_{Search} = a_{Retract} = v_{I++DME}$$

Ferner wird der maximal zulässige Ruck als Erfahrungswert auf das 10fache des einheitslosen Betrages der maximal zulässigen Beschleunigung gesetzt.

$$j_{max} = 10 \cdot \|a_{max}\| \tag{4.2}$$

Darüber hinaus stellt I++DME weder Befehle zum Aktivieren noch zum Deaktivieren der Antriebssysteme bereit. Da diese jedoch eine zentrale Funktion von NPM-Maschinen darstellen, muss eine andere Möglichkeit gefunden, um dem Anwender den Zugriff auf die Antriebe zu ermöglichen. Für die Realisierung dieser Funktion kann auf die I++DME-Sensordatenbank zurückgegriffen werden. Durch das Einfügen eines standardmäßig vorhandenen Pseudosensors, welcher den deaktivierten Zustand signalisiert und steuert, kann der Anwender die Antriebssysteme deaktivieren und mittels des I++DME-Befehls zum Anfahren der Nullposition wieder in den aktivierten Zustand versetzen. Ferner kann dem Anwender auch eine unvorhergesehene Deaktivierung der NPM-Maschine signalisiert werden. Registriert die Steuerungssoftware eine unvorhergesehene Deaktivierung, wird der Anwender durch einen asynchronen Sensorwechsel auf den Pseudosensor über das Abschalten der Antriebe informiert.

4.4.1.2 Adapter für die Maschinenschnittstelle

In Anlehnung an Kapitel 2, in welchem die Schnittstellen beider NPM-Maschinen-Generationen dargestellt und auf ihre Unterschiede eingegangen wurde, wird der erforderliche Funktionsumfang und der Aufbau der Adapter-DLLs im Folgenden dargelegt. Die NPM-Maschinen verwenden unterschiedliche Datentransportmedien und Kommunikationsmechanismen für die Übertragung von Messdaten und implementieren die ihre Schnittstellenbefehle mit identischem Funktionsumfang aber unterschiedlichen Aufrufparametern. Die Abbildung dieser Unterschiede auf die gemeinsame Positionierschnittstelle zur Hardwareabstraktionsebene obliegt den Plugin-DLLs.

NMM-1 Plugin

Das Steuerungsplugin für die NMM-1 besitzt einen sehr geringen Funktionsumfang, wodurch die interne Struktur des Adapters sehr einfach gestaltet werden kann. Wie in Abbildung 34 dargestellt beschränkt sich der Aufbau des NMM-1 Adapters darauf, den USB-Treiber der NMM-1 zu laden und Aufrufe der Positionierschnittstelle auf direktem Weg an diesen weiterzuleiten. Der USB-Treiber überträgt diese Aufrufe seinerseits in USB-Pakete, welche von der NMM-1 empfangen und verarbeitet werden können.

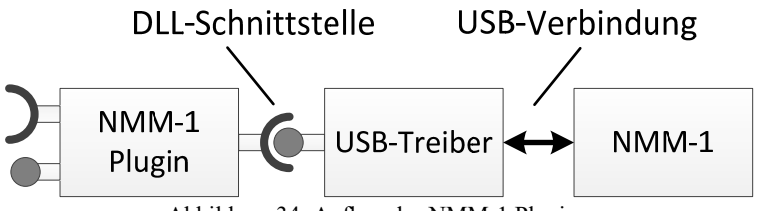


Abbildung 34: Aufbau des NMM-1 Plugins

Da die Steuerschnittstelle des NMM-1 Treibers nicht die komplette Funktionalität der Positionierschnittstelle abdeckt, werden die bereitgestellten Befehle zusätzlich um die Explorationsfunktionen und die statischen Adressierungsparameter sowie die Rückrufschnittstellen für die Messdaten- und Fehlerübertragung ergänzt.

Da der USB-Treiber der NMM 1 von sich aus keine Funktionsrückrufe unterstützt, müssen diese durch die Adapter DLL bereitgestellt werden. Dazu werden, nachdem die DLL durch die zentrale

Steuerungssoftware in den Speicher geladen wurde, mehrere Threads gestartet. Jeder Thread besitzt als leichtgewichtiger Prozess einen eigenen, nebenläufigen Kontrollfluss, wodurch es möglich ist, parallel zur Datenverarbeitung der Steuerungssoftware Zustandsänderungen wie eine unvorhergesehene Deaktivierung der Antriebe, und Messdaten von der NMM-1 abzufragen und über die Rückruffunktionen an die Steuerungssoftware zu übermitteln.

Um vollständig ausschließen zu können, dass gleichzeitig zwei Prozesse auf das NMM-1 Positionierplugin zugreifen und um Störungen im Messablauf und mögliche Kollisionen zu vermeiden, wird der Zugriff auf die DLL über eine *Critical Section* [74] verriegelt. Die *Critical Section* ist ein softwaretechnischer Mechanismus, um nebenläufige Prozesse zu synchronisieren und den Zugriff auf Ressourcen zu steuern. *Critical Sections* werden als betriebssysteminterner Mechanismus vom Windows Application Programming Interface (kurz: WinAPI) bereitgestellt. Um sicher zu stellen, dass stets nur ein Anwender den Zugang zur Steuerung der NPM-Maschine erhält, wird vor dem Aufbau einer USB-Verbindung zur NMM-1 geprüft, ob die Ressource frei ist und erst dann der Zugang bereitgestellt. Bis zum Abbau der USB-Verbindung erhält der eingetretene Prozess das alleinige Nutzungsrecht über das Positioniersystem.

NPMM-200 Plugin

Die zweite Option zur Ansteuerung von NPM-Maschinen stellt der Steuerungsadapter für Positioniersysteme des Typs NPMM-200 bereit. Das Plugin kapselt gemäß dem Schnittstellenkonzept alle erforderlichen Datenverarbeitungsmechanismen, um die Funktionalität einer NPMM-200 auf die Positionierschnittstelle abzubilden. Da NPM-Maschinen vom Typ NPMM-200, im Gegensatz zur NMM-1, Messdaten, Zustandsänderungen und Fehler ereignisgetrieben über mehrere logische Verbindungen übertragen und zudem die leistungsfähigere Datenübertragungstechniken TCP/IP über Ethernet verwenden, ist der Aufbau des Adapters der NPMM-200 komplexer als für eine NMM-1.

Der Adapter muss über die Socket API des Betriebssystems für jeden Kanal eine TCP-Verbindung zur NPMM-200 aufbauen. Zudem sind Algorithmen erforderlich, um Befehle, Messdaten und Zustandsänderungen zwischen der funktionsorientierten Positionierschnittstelle und der datenstrom-basierten TCP-Kommunikation zu übertragen. Neben der Transkription von Aufrufbefehlen sind insbesondere die asynchron eintreffenden Messdaten aller Teilsysteme der NPMM-200 zu synchronisieren. Wie in Abbildung 35 dargestellt, müssen dazu alle eintreffenden Datenpakete zunächst kanalweise in einem Empfangsschieberegister (first-in first-out buffer, kurz: FIFO) zwischengespeichert werden, bis alle Schieberegister zumindest einen Datensatz enthalten. Ist diese Bedingung erfüllt, werden die Datensätze am Beginn jeder FIFO entnommen, entsprechend der Anwendervorgaben verarbeitet und zu einem schnittstellenkonformen Datenpaket kombiniert, welches über die Messdatenrückruffunktion an die Steuerungssoftware zu übertragen ist. Darüber hinaus funktionieren die FIFO-Register als adaptive Messdatenpuffer und ermöglichen intrinsisch den Ausgleich von Lastspitzen, wodurch einem Messdatenverlust im Falle temporärer Überlastung der Verarbeitungsalgorithmen vorgebeugt wird.

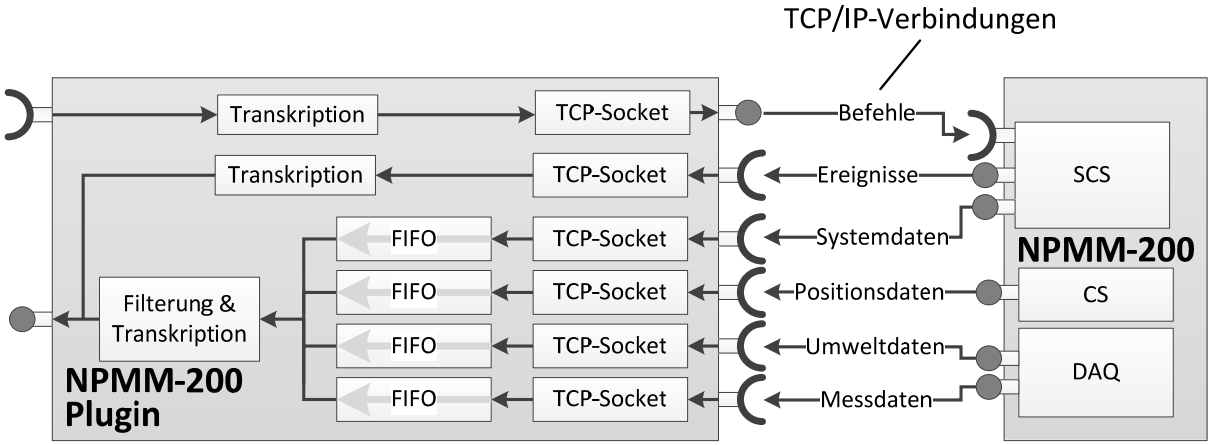


Abbildung 35: Aufbau des Schnittstellenadapters der NPMM-200

Hardware-Emulations Plugin

Für die entwicklungsbegleitende Validierung des Demonstrators ist es zweckmäßig, die Steuerungssoftware auch ohne Positionierhardware in Betrieb nehmen zu können. Aufgrund der starken Ausrichtung der Steuerungssoftware auf Schnittstellen und die dadurch erreichte Hardwaretransparenz ist es ohne weiteres möglich, anstatt eines Hardwareadapters einen Teststummel als Hardwaresimulator zu verwenden. Die Schnittstelle abstrahiert das vom Adapterplugin eingebundene System vollständig, weshalb es unerheblich ist, ob tatsächlich eine reale NPM-Maschine eingebunden wird oder ob die Abläufe durch ein Programm simuliert werden. Dementsprechend ist das Hardware-Emulator Plugin ein Softwareadapter, welcher die erforderlichen Algorithmen für die Simulation einer NPM-Maschine bereitstellt und Systemzustände, Zustandstransitionen und die Messdatenerfassung für entwicklungsbegleitende Tests in ausreichendem Maße nachbilden. Der Fokus des Emulators liegt auf der Nachbildung von Messabläufen und der damit verbundenen Erzeugung von Messdaten.

Um mit der Steuerungssoftware Einfluss auf die simulierten Messabläufe nehmen zu können, ist ein parametrierbarer Bahngenerator erforderlich (siehe Abbildung 36), welcher realitätsnahe Messdaten simuliert. Da alle Mess- und Bewegungsbefehle der Positionierschnittstelle auf Linien- oder Kreisbahnen basieren, bilden diese Bewegungstypen die Basis des Bahngenerators. Seine Arbeitsweise beruht auf der Stapelverarbeitung der beliebig miteinander kombinierbaren Mikrobewegungen. Wird die Ausführung einer komplexen Messung oder Bewegung über die Positionierschnittstelle veranlasst, wird ein Stapel von Mikrobewegungen erzeugt, welche in Folge seiner sequentiellen Abarbeitung den makroskopischen Bewegungsablauf nachbildet. Während der Abarbeitung der Mikrobefehle werden die Systemzustände (z. B. aktuelle Position und Geschwindigkeit) im Zustands- und Parameterspeicher des Simulators kontinuierlich aktualisiert. Handelt es sich bei dem Aufruf um einen Messbefehl, sind neben der fortlaufenden Aktualisierung zusätzlich Messdaten entlang der simulierten Trajektorie zu generieren und an die Steuerungssoftware zu übertragen. Da dem Emulator keine Oberfläche für die Antastung zur Verfügung steht, können Freiformscans nur eingeschränkt simuliert werden. Es ist zweckmäßig, bei diesen Befehlen eine Gerade bzw. einen Kreisbogen zwischen Start- und Zielpunkt als Trajektorie anzunehmen.

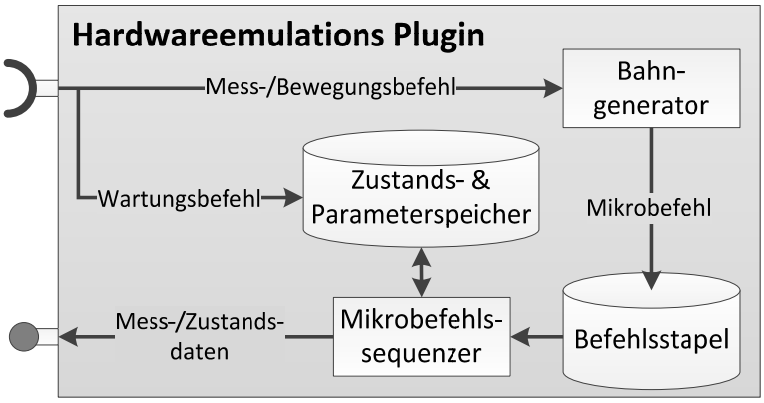


Abbildung 36: Aufbau des Hardware-Emulator Plugins

Die Positionierschnittstelle verfügt neben Mess- und Positionierbefehlen auch über diverse Wartungsfunktionen, mit welchen globale Systemparameter für das Positioniersystem oder die taktilen Sensoren abgerufen und eingestellt werden können. Diese Funktionen werden von der Steuerungssoftware nur intern verwendet und ihre Parametrierung wirkt sich nicht soweit auf Mess- und Positionierverhalten der NPM-Maschine aus, dass ihre Daten in der Simulation Berücksichtigung finden müssen. Der Emulator soll zwar das Setzen und Auslesen der damit verbundenen Daten ermöglichen, aber keine Validitätsprüfung oder Weiterverarbeitung der Daten im Rahmen der Simulation durchführen.

4.4.2 Schnittstellen für optische Sensoren

Die konsistente Integration optischer Sensoren in das Steuerungskonzept ist für die Nanometrologie von großer Bedeutung. Die strikte Ausrichtung des Steuerungsprotokolls I++DME auf taktile Sensoren ist für die Integration dieser Sensorklasse jedoch hinderlich. Optische Sensoren sind im Umfeld der makroskopischen KMT eine vergleichsweise neue Entwicklung, welche im Gegensatz zur Nanometrologie nur eine untergeordnete Bedeutung besitzt. Zwar existieren leistungsfähige Konzepte, wie der I++DME Erweiterungsstandard OSIS, welcher ihre Integration in ein I++DME Umfeld ermöglichen jedoch ist auch OSIS nicht in der Lage, die strikte I++DME-Begrenzung auf Einzelpunktantastungen zu überwinden. Dies kann nur durch eine Erweiterung von Befehlsschnittstelle und Datenverarbeitungslogik erreicht werden.

Trotz der bestehenden Beschränkungen auf Punktmessungen ist der Integrationsansatz über OSIS dennoch eine interessante Alternative. Durch die damit gewährleistete Schnittstellenkompatibilität ist es möglich, mit Kombination konventioneller makroskopischer Anwendersoftware auch mit optischen Sensoren Messungen durchzuführen. Um eine weitgehende Kompatibilität mit bestehender Anwendersoftware bei gleichzeitiger Erweiterung der Funktionalität zu ermöglichen und darüber hinaus dem Anwender dennoch die volle Funktionalität optischer Sensoren für Messungen zur Verfügung zu stellen, werden im weiteren Verlauf beide Ansätze zur Integration optischer Sensoren in die Steuerungssoftware verfolgt.

4.4.2.1 OSIS als I++DME Erweiterungsschnittstelle für optische Sensoren

Optical Sensor Interface Standard

Etwa seit der Jahrtausendwende verfügen Datenverarbeitungssysteme über eine hinreichende Rechenkapazität, um die Rohdaten bildgebender Verfahren in akzeptabler Zeit für metrologische Anwendungen auswerten zu können. Die Möglichkeiten und Vorteile optischer Sensoren für die KMT wurden schnell erkannt und es gab erste Bestrebungen, diese Sensoren mit KMGs zu nutzen. Dennoch konnten sich optische Sensoren bisher nicht flächendeckend etablieren. Als Ursache werden einerseits

die unübersichtliche Marktlage verfügbarer Sensoren und andererseits ein Mangel an Vergleichbarkeit ihrer Leistungsfähigkeit aufgrund fehlender Standards angeführt. Die Hauptursache liegt jedoch in den fehlenden Richtlinien für die Integration in ein KMG [75]. Bisherige Umsetzungen sind in der Regel nicht portierbare und kundenspezifische Einzelanfertigungen.

Keferstein [76] schätzt die Kosten für die Integration eines optischen Sensors in vergleichbarer Höhe wie der Anschaffungspreis eines neuen KMGs ab. Die hohen Kosten stellen ein großes Hindernis für die Verbreitung optischer Sensoren dar. Zudem existiert keine Richtlinie, um die Fähigkeiten und Leistungsparameter verschiedener optischer Sensoren zu vergleichen, weshalb viele Anwender von der Nutzung optischer Sensoren absehen. Um optische Sensoren für eine breite Anwenderschaft verfügbar zu machen und die Integrationskosten zu minimieren, wurde die Initiative OSIS - ein Zusammenschluss verschiedener KMG-Hersteller, Hersteller optischer Sensoren und der Interstaatlichen Hochschule für Technik Buchs - zur Erarbeitung umfassender Richtlinien für die Interoperabilität optischer Sensoren mit KMGs ins Leben gerufen. In ihrem Fokus steht die Definition von geeigneten Normen und Schnittstellen für die transparente Interoperabilität von KMG und optischem Sensor [31]. Wegen der umfangreichen Problematik ist das Gremium in drei Arbeitsgruppen aufgeteilt:

- Arbeitsgruppe 1 „Mechanische und elektrische Schnittstellen”
- Arbeitsgruppe 2 „Datenintegration“
- Arbeitsgruppe 3 „Leistung, Spezifikation und Kalibrierprozesse”

Für die Integration optischer Sensoren in die I++DME Schnittstelle ist nur die Arbeitsgruppe 2 von Relevanz. Die beiden anderen Arbeitsgruppen definieren Hardwareschnittstellen (Arbeitsgruppe 1) bzw. fassen Richtlinien und Normen für standardisierte Vergleichsmethoden (Arbeitsgruppe 3) zusammen. Auch wenn die Festlegungen der Arbeitsgruppe 1 aus der makroskopischen Metrologie kaum auf NPM-Maschinen übertragbar sind, bilden die Festlegungen der übrigen Arbeitsgruppen adaptierbare Grundlagen für die informationstechnische Integration und die Vergleichbarkeit optischer Sensoren.

OSIS-Integrationsmodell

Die Arbeitsgruppe 2 legt für die informationstechnische Integration verschiedene logische Schnittstellen zwischen Sensor und KMG [77] fest, welche in Abbildung 37 dargestellt sind. Die Übersicht visualisiert alle im Zusammenhang mit der Bedienung und Steuerung eines KMG relevanten Schnittstellen und ergänzt diese durch die von OSIS festgelegten Schnittstellen für optische Sensoren. Die blau dargestellten Schnittstellen sind nicht bzw. nur mittelbar Bestandteil von OSIS. Um unnötige Festlegungen zu vermeiden und den aktuellen Stand der Technik in den Entwurf zu integrieren, wird von OSIS so weit wie möglich auf etablierte Standards für die Kommunikation zwischen den Schnittstellen zurückgegriffen.

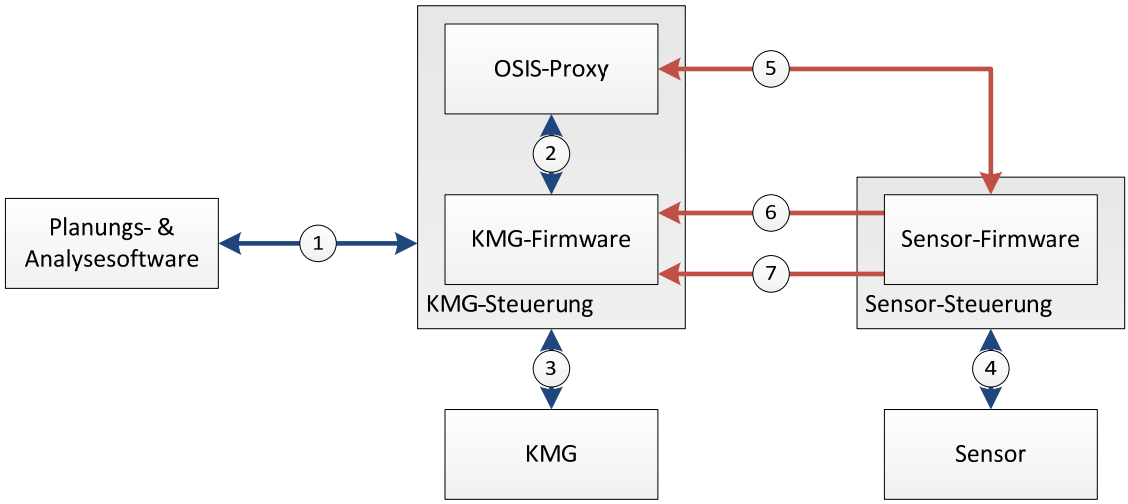


Abbildung 37: OSIS Integrationsmodell nach [17] (rot: OSIS-Schnittstellen, blau: weitere Schnittstellen)

So wird für die Speicherung und den Entwurf der Prüfplanung auf die Norm ISO 22093 (Dimensional Measurement Interface Standard) zurückgegriffen und das I++DME Protokoll für die Kommunikation zwischen der Anwendersoftware und dem KMG (Schnittstelle 1) genutzt. Die nicht in den Geltungsbereich von OSIS fallenden Schnittstellen 2 bis 4 sind weder für den Nutzer noch die OSIS-Komponenten zugänglich. Sie dienen dem hardwarenahen Datenaustausch und der Steuerung des Positioniersystems. Diese Schnittstellen sind vollständig vor den Komponenten zur Steuerung optischer Sensoren verborgen und nur der Vollständigkeit halber dargestellt. Im OSIS-Standard wird lediglich auf die Schnittstellen 6 bis 7 Bezug genommen.

Die Festlegung der Schnittstelle 5 ist das Hauptbetätigungsfeld der OSIS Arbeitsgruppe 2. Da alle Funktionalitäten entsprechend der OSIS-Festlegungen in der Sensorfirmware zu implementieren sind, dient diese Schnittstelle der Trennung von KMG und den Sensorkomponenten. Ferner stellt diese Schnittstelle den wichtigsten Kommunikationspfad der gesamten OSIS-Architektur dar. Alle nicht zeitkritischen Daten und Befehle, wie Messdaten und -befehle werden darüber ausgetauscht. Die Steuerung der Kommunikation zwischen Sensor-Firmware und OSIS-Proxy obliegt entsprechenden Modulen des OSIS-Proxy.

Die Trigger-Schnittstelle (Schnittstelle 6) ist eine einfache elektrische Koppelung von Sensor und KMG. Durch ein vom Sensor ausgelöstes elektrisches Signal wird von Sensor und KMG eine parallele Datenaufzeichnung veranlasst. Das KMG speichert die aktuellen Achsenpositionen und der Sensor ihm vorliegende Rohdaten bzw. Koordinaten der Objektoberfläche. Um die eindeutige Zuordnung der dezentral gehaltenen Messdaten zu ermöglichen, werden die Triggerimpulse in beiden Modulen mitgezählt und die Messdaten damit indiziert.

Um der Objektoberfläche folgen zu können, ist es bei einigen optischen Sensoren notwendig, die Positioniereinheit während eines Scanvorgangs nachzuführen (Schnittstelle 7). Damit das Nachführen mit einer möglichst geringen Verzögerung erfolgt, wird der dafür erforderliche Datenaustausch über die Feedback-Schnittstelle abgewickelt.

Die Leistungsfähigkeit von OSIS für die Nanometrologie soll im praktischen Einsatz untermauert werden. Dazu soll der Demonstrator der NPMM-Steuerungssoftware im Folgenden um eine OSIS-konforme Schnittstelle erweitert werden.

Integrationskonzept für eine Erweiterung nach OSIS

Die Eingliederung der OSIS-Schnittstelle in das bestehende Konzept erfolgt als eine transparente Metaschicht in der Hardwareebene. Die Metaschicht stellt eine vorgeschobene Schnittstellenebene dar

(siehe Abbildung 38), welche bei der Nutzung optischer Sensoren als Weiche und Multiplexer für Befehlssequenzen bzw. Messdaten fungiert. So kann hardwareseitig eine Auftrennung der Datenpfade für Oberflächenmessdaten und Positionsdaten erreicht werden, wodurch es möglich ist, eine optionale Schnittstelle für optische Sensoren bereitzustellen. Die über beide Pfade eintreffenden Daten werden von der OSIS-Erweiterung gesammelt und zu schnittstellenkompatiblen Messdatenpaketen gebündelt.

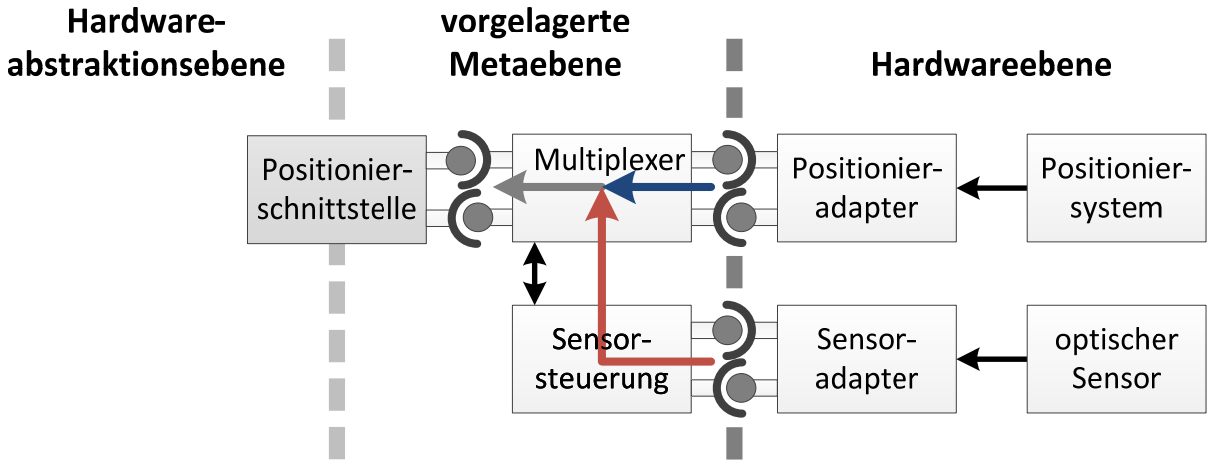


Abbildung 38: Konzept der vorgeschobenen Positinierschnittstelle

Da im Rahmen optischer Antastungen nur eine Teilmenge der Schnittstellenfunktionen verwendet wird, kann die vorgelagerte Metaschicht zwischen Steuerungssoftware und Positioniersystem als "man in the middle" (kurz: MITM)[2] auftreten. Dazu implementiert die OSIS-Erweiterung die Positionierschnittstelle doppelt, wobei sie hardwareseitig mit der reduzierten Schnittstelle als Befehlsquelle und Datensenke bzw. anwenderseitig mit der vollen Schnittstellenimplementierung als Datenquelle und Befehlssenke auftritt.

Die Aufspaltung des Daten- und Befehlsflusses sowie die konsequente Fortführung des Schnittstellenkonzeptes garantiert weiterhin die strikte Trennung von generischer Steuerungssoftware und hardwarespezifischer Maschinensteuerung. Die Module der Hardwareebene existieren weiterhin losgelöst vom Aufbau der Hardwareabstraktionsebene, wodurch sich Änderungen innerhalb einer Ebene nicht auf benachbarte Ebenen auswirken und die vollständige Austauschbarkeit der Hardware über entsprechende Adapter gewährleistet ist.

Konkretisierung des Multiplexers für optische Sensoren

Die Definitionen der OSIS-Arbeitsgruppe 2 legen ein sehr konkretes Fundament für die architektonische Gestaltung der Steuerungskomponente für optische Sensoren. Als weiterer Schritt wird durch die Arbeitsgruppe die klassenweise Zuordnung der OSIS-Architektur auf die Knoten des OSIS-Integrationsmodell festgelegt, wodurch eine Umsetzung der OSIS-Architektur als verteilte Softwarekomponente nahe gelegt wird. Von zentraler Bedeutung für die OSIS-Architektur sind der OSIS-Proxy und die OSIS-Sensorsteuerung. Beide Klassen besitzen eine Referenz auf ihren Partner, mit deren Hilfe sie den Informationsaustausch über dessen öffentliche Methoden durchführen.

Das von OSIS vorgegebene Verteilungskonzept muss für den weiteren Entwurf mit dem Konzept der vorgeschobenen Positionierschnittstelle in Einklang gebracht werden. Dazu wird zunächst die

[2] Das Konzept des MITM ist dem Bereich der Netzwerksicherheit entlehnt, wo es zur Beschreibung eines Angriffstypus auf Rechnersysteme oder -netzwerke Anwendung findet. Der MITM-Angreifer schiebt sich dabei unbemerkt zwischen zwei Interaktionspartner, wodurch ihm das Abhören von Daten oder die gezielte Manipulation von Datenströmen ermöglicht wird.

Verteilung aller Softwarekomponenten einer NPM-Maschine in Abbildung 39 betrachtet. Die bereits bekannte Sensorsteuerung wird auf ein dediziertes Datenverarbeitungssystem ausgelagert und über die neu eingeführte Sensorschnittstelle mit dem Rohdatenerfassungssystem kommuniziert. Ferner konvertiert sie die an der Objektoberfläche erfassten Daten in metrologisch verwertbare Raumkoordinaten, welche über die eingebettete Referenz der OSIS-Steuerung an den OSIS-Proxy des Messdaten- und Positionsmultiplexers auf dem Datenverarbeitungssystem der zentralen NPMM-Steuerungssoftware übermittelt werden.

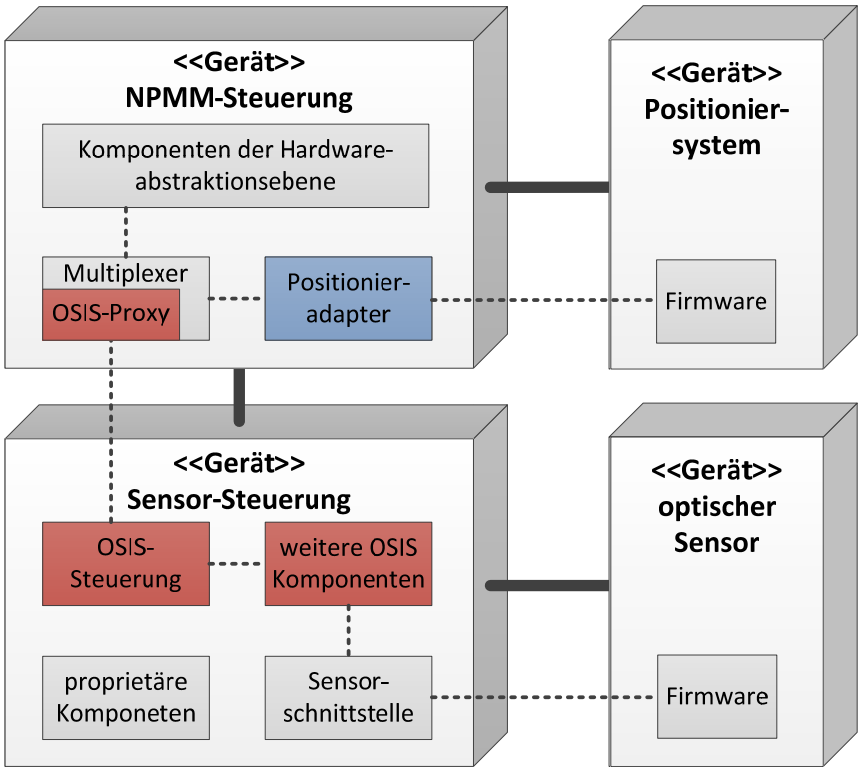


Abbildung 39: Verteilung der Komponenten der OSIS-Schnittstelle 3
(blau: Steuerungskomponenten des Positioniersystems, rot: Komponenten für die Steuerung optischer Sensoren)

Desweiteren soll die I++DME-konforme Erweiterung für optische Sensoren so umgesetzt werden, dass es keine Rolle spielt, wie die tatsächliche Verteilung der Komponenten auf verschiedene Plattformen erfolgt. Zwar kann davon ausgegangen werden, dass die Komponenten zur Steuerung des optischen Sensors perspektivisch in ein eingebettetes System ausgelagert werden, im Rahmen der Entwicklung muss es jedoch auch möglich sein, die Steuerung auf einem PC-System, gegebenenfalls sogar auf dem System der zentralen NPMM-Steuerung auszuführen.

Um Methodenaufrufe über die Grenzen eines Datenverarbeitungssystems hinweg in einem Netzwerk zu ermöglichen, gibt es neben der aufwändigen und fehleranfälligen Implementierung proprietärer Verfahren verschiedene etablierte Rahmenwerke. Durch ihren Einsatz kann der Entwurfsprozess beschleunigt und die zu implementierende Software effizient und ausfallsicher gestaltet werden. Da sich diese Grundgerüste jedoch zum Teil sehr stark in ihren Eigenschaften unterscheiden, wird anhand der folgenden Anforderungen das optimal geeignete Framework ausgearbeitet.

1. Die zu entwickelnde Software folgt den Paradigmen der objektorientierten Programmierung, weshalb dieses Konzept auch für den Datenaustausch beibehalten werden soll.
2. Das Framework soll in seiner Verwendung vollkommen transparent sein. Für den Anwender darf kein Unterschied in der Nutzung lokaler Komponenten zu Komponenten auf einem anderen Rechnersystem erkennbar sein.

3. Das Framework soll unabhängig von Programmiersprache und Betriebssystem sein, um die zukünftige Sensorentwicklung nicht auf Sprachen oder Plattformen einzuschränken.
4. Der Datenaustausch soll schnell und effizient erfolgen. Hohe CPU- oder Netzwerklast aufgrund der Kommunikation zwischen verteilten Komponenten ist nicht akzeptabel, da so der Durchsatz bei der Verarbeitung metrologischer Daten reduziert wird.

Die hier genannten Anforderungen sind für die schnelle und flexible Integration von OSIS von zentraler Bedeutung. Die Objektorientierung folgt als logische Konsequenz der Festlegungen von I++DME und OSIS, wohingegen die Sprachen- und Plattformunabhängigkeit nicht zwingend aus den Standards selbst erwächst. Sollen Sensorsysteme jedoch zukünftig auf einem eingebetteten System mit nicht näher spezifizierten Betriebssystemen integriert werden, ist die Erfüllung dieser Punkte unumgänglich. Eingebettete Systeme haben im Vergleich zu konventionellen PC-Systemen viele Vorteile. Diese Systeme sind kompakt, hoch integriert, energiesparend und weniger störanfällig. Seit einigen Jahren integrieren Kamerahersteller neben den Sensoren und der Ausleseelektronik zunehmend leistungsfähige Rechentechnik in ihre Kamerasysteme. Diese eingebetteten Smart-Kameras verfügen neben digitalen Signalprozessoren für sehr umfangreiche Bildverarbeitungsaufgaben zum Teil zusätzlich auch über Paralleldatenverarbeitungsprozessoren. Die Systeme QT-APX Camera Plattform von Qtechnology A/S [78] oder die XCI-Serie „Intelligent Cameras“ von Sony [79] sind nur zwei Beispiele dafür. Diese eingebetteten Kamerasysteme sind in der Lage, die komplette Bildverarbeitung auf einer Hardware auszuführen und statt der erfassten Rohbilddaten bereits abstrakte Bildmerkmale oder gar qualitative Aussagen und Daten auszugeben. Dadurch entfällt die Notwendigkeit, eine breitbandige Verbindung zwischen dem Sensor und dem Bildverarbeitungs-PC herzustellen.

Um die Flexibilität von Smart-Kameras weiter zu steigern, können Betriebssysteme (Windows Embedded [80] oder µClinux [81]) auf den Systemen installiert werden [82]. Diese Betriebssysteme wurden speziell für eingebettete Rechentechnik entwickelt und ermöglichen die abstrahierte Nutzung der zum Teil sehr speziellen Hardwareschnittstellen. Sie erlauben so den Einsatz verschiedener Hardwareplattformen und ermöglichen eine schnelle und einfache Nutzung standardisierter Kommunikationsmedien wie Ethernet, USB oder FireWire durch die Verwendung der in das Betriebssystem integrierten Schnittstellen. Insbesondere die zunehmende Verwendung des Gigabit-Ethernet-Standards als schnelle Datenübertragungsschnittstelle für Kameras und Smart-Kameras bildet eine ideale Grundlage. Generell werden von allen Rahmenwerken für verteilte Softwareentwicklung Sockets [83] für die Kommunikation verwendet, welche wiederum eine native Kommunikationsplattform der eingebetteten Betriebssysteme darstellen, wodurch ihre Anwendung in eingebetteten Systemen auf elegante und einfache Weise ermöglicht wird.

Die wichtigsten Vertreter der Rahmenwerke für verteilte Software werden im Folgenden näher erläutert. Die Eigenschaften der Programmiergerüste unterschieden sich zum Teil deutlich, weshalb sie den hier gestellten Anforderungen gegenübergestellt werden und abschließend anhand dieses Vergleichs das bestgeeignete Framework für die Implementierung der OSIS-Schnittstelle ausgewählt wird.

4.4.2.2 Grundgerüste für verteilte Programmierung

Remote Procedure Call

Remote Procedure Call (engl. kurz RPC) ist der älteste, im Zusammenhang mit verteilten Softwaresystemen verwendete Mechanismus. Die erste Beschreibung von Aufrufen verteilter Prozeduren stammt von James E. White aus dem Jahr 1976 [84]. Die grundlegende Idee von PRC besteht darin, für Aufrufe entfernter Prozeduren und Funktionen die gleichen Prinzipien wie bei einem

lokalen Funktionsaufruf zu verwenden. Ein aufrufender Prozess übergibt Eingabeparameter an einen zweiten aufgerufenen Prozess. Der aufgerufene Prozess arbeitet, während der aufrufende Prozess wartet. Beendet der aufgerufene Prozess die Datenverarbeitung, übergibt er die Rückgabeparameter an den aufrufenden Prozess, welcher seine Arbeit fortsetzt. Diese Vorgehensweise entspricht einem synchronen verbindungsorientierten Kommunikationsprotokoll und ist allen Frameworks für verteilte Programmierung gemein.

Die Kommunikation und Synchronisation wird von einer Zwischenschicht bewirkt. Für die korrekte Arbeit der Zwischenschicht ist es erforderlich, dass für eine Prozedur vom Programmierer neben der implementierten Semantik eine abstrakte Beschreibung ihrer Schnittstelle bereitgestellt wird. (siehe Abbildung 40). Der Programmierer muss lediglich dafür Sorge tragen, dass dem Grundgerüst die auszuführende Funktionalität bereitgestellt wird.

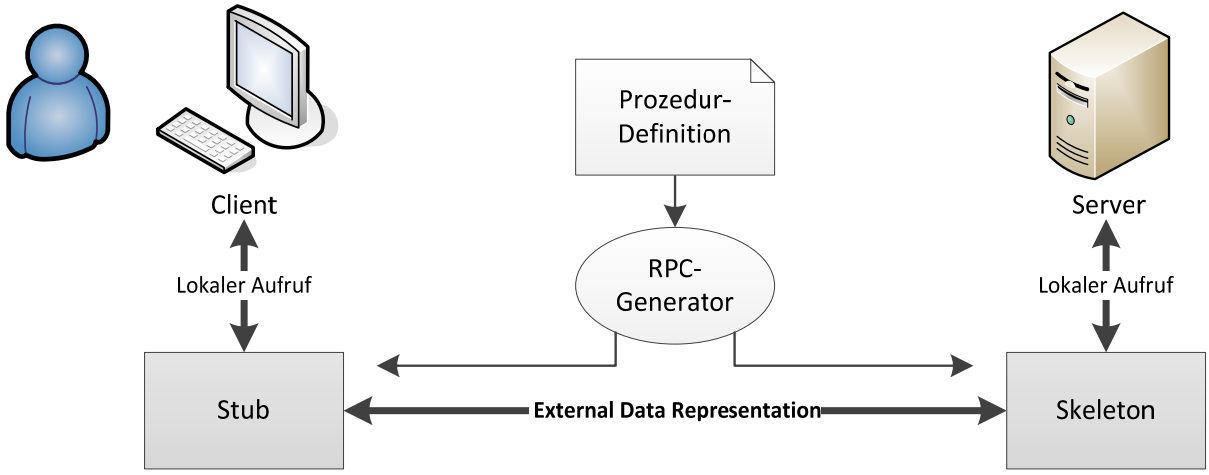


Abbildung 40: Funktionsweise eines Remote Procedure Calls

Aus einer Prozedurdefinition erzeugt ein RPC-Generator zwei Kommunikationsstummel (Stub & Skeleton), welche über alle erforderlichen Mechanismen für die Netzwerkkoordination und die Synchronisation von aufrufendem (lokalem) und aufgerufenem (entfernten) Prozess verfügen.

Die beiden Kommunikationsstummel werden wie folgt auf die Datenverarbeitungshardware verteilt. Der Stub befindet sich auf dem Anwendungsrechner und dient der Übermittelung lokaler Aufrufe über das Netzwerk hinweg an den Skeleton. Dieser befindet sich auf dem Server und stellt die eigentliche Verbindung zu den Datenverarbeitungsfunktionen zur Verfügung. Soll eine auf dem Server befindliche Prozedur auf dem Anwendungsrechner verwendet werden, ruft das Programm eine lokale Stellvertreterprozedur des Stub auf. Der als Proxy agierende Stub verpackt die Aufrufparameter in das External Data Representation Format und sendet die Anfrage an den entfernten Skeleton. Der Skeleton entpackt die Aufrufparameter, führt die eigentliche Prozedur aus und übermittelt nach Abschluss der Verarbeitung die Rückgabeparameter mit demselben Mechanismus an den Stub.

Der Programmierer kann somit via RPC lokale und entfernte Funktionen transparent verwenden. Auch wenn das reine RPC Verfahren aufgrund der fehlenden Unterstützung der Objektorientierung kaum mehr von praktischer Bedeutung ist, bildet es die Grundlage der meisten höheren Frameworks [85]. Bis auf die Forderung nach Objektorientierung werden alle Eigenschaften durch das RPC-Konzept erfüllt (siehe Tabelle 12). Da eine konsistente Überführung prozeduraler Aufrufe auf objektorientierte Methodenaufrufe mit erheblichen Schwierigkeiten verbunden ist, kommt RPC allerdings für die Umsetzung von OSIS nicht in Frage.

Pro	Contra
plattformunabhängig	keine Objektorientierung
geringer Overhead	
transparent	
einfache Anwendung	

Tabelle 12: Eigenschaften von RPC

Remote Method Invocation

Remote Method Invocation (engl. kurz RMI) ist die Weiterentwicklung der RPC hin zu den Paradigmen der objektorientierten Programmierung [86]. Bei RMI werden im Gegensatz zu RPC nicht Prozeduren, sondern öffentliche Methoden von serverseitigen Klasseninstanzen bereitgestellt. Wie bei RPC erfolgt auch bei RMI die Erzeugung von Stub und Skeleton aus einer abstrakten Beschreibung heraus. Analog erfolgt auch die Nutzung der beiden Kommunikationsstummel. Der bekannteste Vertreter für ein RMI-Framework ist JAVA RMI [87]. Die RMI-Eigenschaften mit Relevanz für die Implementierung der OSIS-Erweiterung sind in Tabelle 13 aufgelistet.

Pro	Contra
plattformunabhängig	sprachenabhängig (nur in Java verfügbar)
Objektorientierung	hohe CPU-Last (Java Interpreter notwendig)
transparent	

Tabelle 13: Eigenschaften von RMI

Common Object Request Broker Architecture

Eine konsequente Weiterentwicklung der RMI stellt die Common Object Request Broker Architecture (CORBA) dar. CORBA ist ein von Programmiersprachen und Betriebssystemplattform unabhängiger Industriestandard, der den Paradigmen der Objektorientierung folgt und vollständig plattform- und sprachenunabhängig ist [88]. Die große Variabilität wird durch den Einsatz einer abstrakten Defintionssprache für die Beschreibung der verteilten Komponenten erreicht.

Interface Definition Language

Ähnlich wie bei RPC werden bei CORBA die verteilten Objekte zunächst als abstrakte Definition in Form von Klassenschnittstellen in einer eigenen Programmiersprache, der Interface Definition Language (IDL) festgelegt. Syntax und Umfang der IDL-Datentypen bzw. IDL-Datenstrukturen ähneln stark der Programmiersprache C++, jedoch müssen übergebene Parameter explizit in Eingabe-, Ausgabe- bzw. in bidirektionale Parameter unterschieden werden [89]. Mittels eines IDL-Compilers wird äquivalent zu RPC bzw. RMI aus der Schnittstellendefinition der Quellcode für Stub und Skeleton erzeugt. Erst dieser automatisch erzeugte Quellcode kann in ein Softwareprojekt eingebunden werden.

Aufbau und Funktionsweise

Das zentrale CORBA-Element ist der Object Request Broker (ORB), welcher die Kommunikation zwischen verteilten Objekten ermöglicht. Der ORB nimmt dem Anwender die organisatorischen Standardaufgaben in Bezug auf die Datenübertragung ab. Er lokalisiert die Gegenstelle für eine

Kommunikation und erledigt den Datentransport vollständig transparent. In Abbildung 41 ist der Aufbau des ORB dargestellt.

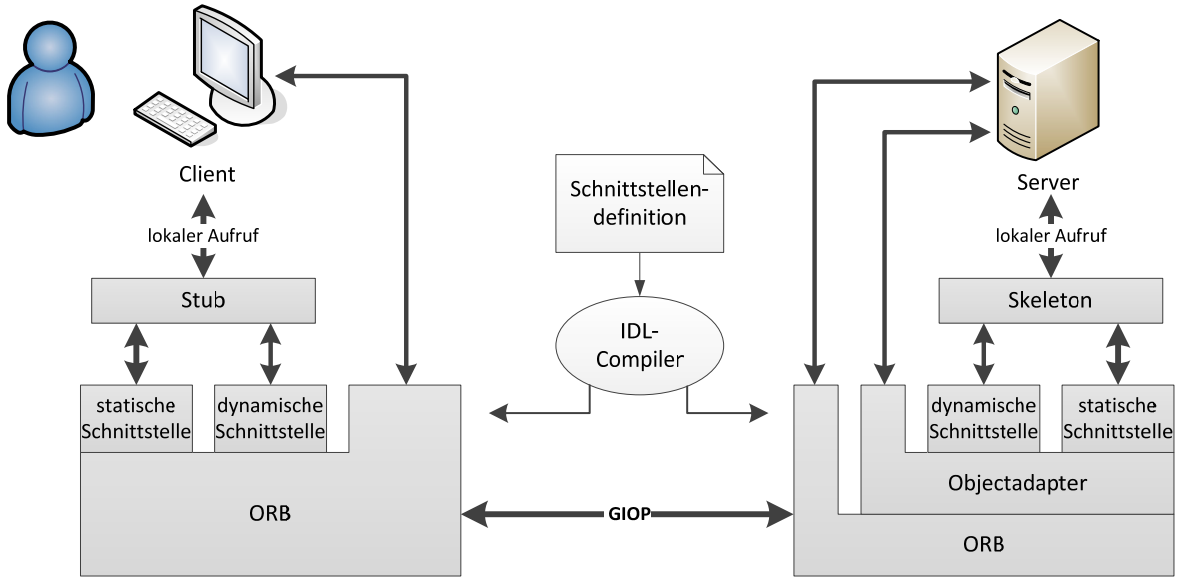


Abbildung 41: Aufbau des Object Request Broker

Analog zu RPC und RMI sind Stub und Skeleton Proxyobjekte für die entfernten Programmteile. Ein vom Client veranlasster Methodenaufruf ist stets an den lokalen Stub gerichtet. Dieser verpackt die Aufrufparameter und leitet den Aufruf über das Static Invocation Interface an den ORB weiter. Ist dem ORB der Standort des Serverobjektes bekannt, leitet er die Anfrage per General Inter ORB Protocol an den serverseitigen ORB weiter. Der entfernte ORB übergibt den Aufruf an den Objektadapter, dieser instanziiert gegebenenfalls das Objekt und reicht den Aufruf über das Static Skeleton Interface an den Skeleton weiter. Der Skeleton entnimmt dem Aufruf die Parameter und ruft damit die betreffende Methode des zuvor instanziierten Objektes auf. Rückgabeparameter und Rückgabewert der Methode werden analog in entgegengesetzter Richtung versandt [90].

Für die Nutzung dynamischer Methodenaufrufe, d. h. Schnittstellen, welche zum Zeitpunkt der Kompilierung nicht bekannt sind, ermöglicht CORBA über das Dynamic Invocation Interface und das Dynamic Skeleton Interface eine Definition zur Laufzeit. Die komplexe dynamische Schnittstellendefinition wird im Rahmen dieser Arbeit allerdings nicht angewandt, da die umzusetzende Architektur a priori festgelegt werden kann.

Da CORBA an sich als ein Standard zur Interaktion verteilter Software verstanden werden kann, gibt es natürlich zahlreiche freie und kommerzielle Umsetzungen (z. B. TAO [91], MICO [92] oder omniORB [93]). Um die Kommunikation zwischen den unterschiedlichen CORBA-Implementierungen zu gewährleisten, werden die verteilten Objekte durch abstrakte, für den Anwender nicht zugängliche Bezeichner adressiert. Da diese Bezeichner stets global eindeutig sind, ist auch in ausgedehnten Netzwerken, wie dem Internet, eine fehlerfreie Interaktion der verteilten Objekte sicher gestellt. Auch wenn die verschiedenen CORBA-Implementierungen geringe Unterschiede insbesondere mit Rücksicht auf die Effizienz der Datenverarbeitung aufweisen, können die allgemeinen Eigenschaften von CORBA entsprechend Tabelle 14 zusammengefasst werden.

Pro	Contra
geringe CPU-Last geringer Netzwerkoverhead Objektorientierung plattformunabhängig sprachenunabhängig transparent schnelle Entwicklung (Mehrfach-)Vererbung möglich	komplizierte Anwendung Zugriff auf verteilte Objekte über Referenzen

Tabelle 14: Eigenschaften von CORBA

Distributed Component Object Model

Das Distributed Component Object Model (DCOM) [94] ist ein freies Framework zur Programmierung von verteilten Anwendungen. DCOM wurde 1996 von der Microsoft Corporation entwickelt und ist eine auf dem Industriestandard Distributed Computing Environment [95] basierende netzwerkbasierte Weiterentwicklung des Component Object Models (COM) [96], [97]. Aufgrund Microsofts marktbeherrschender Stellung für Desktopcomputer und der effizienten Integration des Frameworks in alle Microsoft-Betriebssysteme ab Windows 95 ist DCOM eine ernstzunehmende Alternative für CORBA.

Prinzipiell ist auch DCOM sprachen- und plattformunabhängig. Es ist jedoch offensichtlich, dass DCOM nicht von vornherein als unabhängiges Framework konzipiert wurde. Zum Einen erfolgt die Kommunikation zwischen Anwendung und COM-Objekten mit Funktionszeigern. Folglich haben Programmiersprachen, in denen Zeiger aus Gründen der Programmiersicherheit nicht zur Verfügung stehen (z. B. Java), erhebliche Schwierigkeiten mit der Integration von DCOM [98]. Zum Anderen ist die Plattformunabhängigkeit insofern eingeschränkt, dass für die Registrierung der COM-Objekte nicht auf eine eigenständige Datenbank, sondern auf die Windows Systemregistry zurückgegriffen wird. Es existieren zwar freie Portierungen für andere Betriebssysteme (z. B. COMsecure) [99], jedoch ist CORBA auf Nicht-Microsoftsystemen für die Realisierung von verteilten Anwendungen eher der Standard.

Neben der Spezialisierung auf die Programmiersprache C++ und die Microsoft-Betriebssysteme wird auch durch das fehlende Vererbungskonzept bei der Definition von DCOM-Objekten ersichtlich, dass DCOM eher eine Technologie zur Kapselung existierender Funktionen ist als ein Rahmenwerk für den grundlegenden Aufbau verteilter Anwendungen. Jede DCOM-Schnittstelle wird durch Aggregation und Delegation gebildet. Die Wiederverwendung des Codes ist damit zwar nicht prinzipiell eingeschränkt, jedoch erzwingt die erneute Verwendung eine zusätzliche Kapselungsschicht und somit einen nicht unerheblichen Implementierungsaufwand und eine zusätzlich erhöhte Rechenlast zur Laufzeit. Bei der Anwendungsentwicklung unter Windows kommen die Vorteile von DCOM vor allem in Verbindung mit der Programmiersprache C++ zum Tragen. Die Dauer für die Entwicklung von verteilten Anwendungen wird dadurch drastisch verkürzt und DCOM-Implementierungen sind aufgrund der betriebssystemnahen Integration schnell und leistungsfähig [85].

Die Arbeitsweise von DCOM ist mit der Funktionsweise von CORBA vergleichbar. Auch DCOM baut auf die RPC/RMI auf und erfordert eine abstrakte Schnittstellenbeschreibung, aus welcher die Proxyobjekte für die lokalen Aufrufe erzeugt werden. Im Gegensatz zu CORBA sind jedoch zwei Schnittstellendefinitionen erforderlich. Neben der bekannten IDL wird zusätzlich eine Object-

Description-Language-Beschreibung für die Informationsverarbeitung dynamischer Methodenaufrufe benötigt [100], [101].

Aufgrund von Schwierigkeiten bei der Kommunikation über das Internet, speziell über Firewalls hinweg, wurde DCOM um diverse Protokolle erweitert und später als .NET Remoting [102] in das sprachenunabhängige .NET Framework integriert. Aufgrund des in diesem Zuge neu eingeführten Simple Object Access Protocols benötigt .NET Remoting jedoch mehr Ressourcen als DCOM. Eine Bewertung von DCOM und .NET Remoting ist in Tabelle 15 und 16 zusammengefasst.

Pro	Contra
geringer Overhead	keine Objektorientierung (keine Vererbung)
schnelle Entwicklung	plattformabhängig (Windows)
transparent	Sicherheitsschwachstellen
Rechteverwaltung möglich	

Tabelle 15: Eigenschaften von DCOM

Pro	Contra
kompatibel zu DCOM	erhöhte CPU-Last
Interface-Definition per IDL entfällt	erhöhte Netzwerklast

Tabelle 16: Eigenschaften von .NET-Remoting

Fazit zu den Frameworks für verteilte Programmierung

Die Anforderungen an ein Framework – Objektorientierung, Transparenz, Sprachen- und Plattformunabhängigkeit sowie Effizienz – werden in der Summe nur durch ein CORBA-Framework erfüllt. Alle anderen Rahmenwerke kommen wegen etwaiger Plattformabhängigkeiten bzw. der fehlenden Unterstützung der Objektorientierung für die Implementierung der entfernten OSIS-Komponenten nicht in Frage.

4.4.2.3 Das OSIS-Rahmenwerk

Basierend auf dem ausgewählten CORBA-Framework wird im Folgenden ein OSIS-Grundgerüst entworfen, welches die Nutzung beliebiger optischer Sensoren mit der Steuerungssoftware ermöglicht. In Abbildung 42 sind auf der linken Seite und dunkel hervorgehoben die virtuellen Stellvertreter der realen Sensorbaugruppen dargestellt. Alle anderen Klassen dienen dem Austausch oder der Speicherung von Informationen.

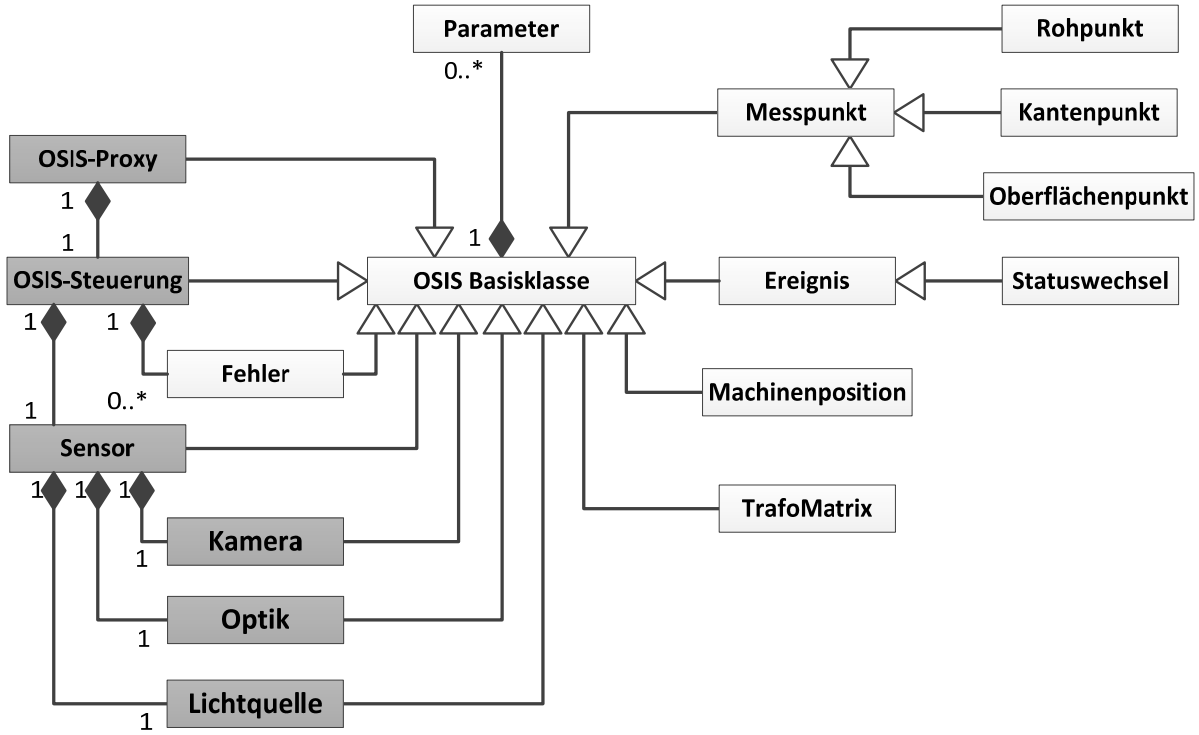


Abbildung 42: Klassendiagramm des OSIS-Grundgerüsts

OSIS wurde als offener Standard mit Rücksicht auf zukünftige Erweiterungen konzipiert. Deshalb werden die Eigenschaften einer Klasse nicht als einfache Werte gespeichert, sondern in einem speziellen Datencontainer, dem Parameter gespeichert. Der Parameter beinhaltet eine kurze Beschreibung der Eigenschaft, ihren Namen, die Typisierung nach OSIS und ihren Datentyp. Um in Parameterobjekten gespeicherte Informationen im OSIS-Rahmenwerk allgemein zugänglich zu machen, wird der Vererbungsmechanismus angewandt.

Alle Klassen außer der Parameterklasse erben die Fähigkeit zum Auslesen und Editieren von Parameterobjekten von der gemeinsamen OSIS-Basisklasse. Aufgrund des offenen Charakters von OSIS kann der Softwareaufbau sensorspezifisch variieren. Aus diesem Grund besitzt die OSIS-Basisklasse – und somit auch alle anderen OSIS-Klassen – zwei interne Listen, in welcher zugehörige Parameterobjekte bzw. OSIS-Objekte hinterlegt werden. Datenzugriffe über das Rahmenwerk sind zunächst an die Parameterliste einer Parameterrepräsentation zu richten. Ist der fragliche Parameter in der Liste aufgeführt, kann er ausgelesen und bearbeitet werden. Anderenfalls wird der Aufruf des anfragenden Objektes zurückgewiesen. Das Zugriffsverfahren auf andere OSIS-Objekte funktioniert äquivalent über die zweite Liste. Auf diese Weise ist es ohne weiteres möglich, die Struktur der Sensorarchitektur beliebig zu skalieren.

Jedoch soll nicht nur die Struktur des Rahmenwerkes variabel sein, sondern auch ein wahlfreier Zugriff der Objekte untereinander über Rechnergrenzen hinweg ermöglicht werden. Wie im letzten Abschnitt erläutert, kommt zu diesem Zweck ein CORBA-Framework zum Einsatz. Dazu werden zunächst die über CORBA zugänglichen Schnittstellen in IDL definiert. Für die Realisierung des OSIS-Rahmenwerkes sind Schnittstellen für die folgenden fünf Klassen hinreichend:

- OSIS-Basisklasse
- OSIS-Proxy
- OSIS-Steuerung
- Messpunkt
- Ereignis
- Sensor
- Parameter

Aus den IDL-Dateien wird mit dem IDL-Compiler der Quellcode der Proxy-Klassen (Stub und Skeleton) erzeugt. Dieser beinhaltet alle notwendigen Mechanismen für die Datenübertragung und die Synchronisation der Aufrufe. Diesem Kommunikationsgrundgerüst wird durch die Vererbung noch die Basisfunktionalität der jeweiligen OSIS Klassen hinzugefügt.

Allerdings kommt es bei diesem Vorgehen bei allen von Messpunkt und Ereignis abgeleiteten Klassen zur Mehrfachvererbung, d. h. die Schnittstelle der OSIS-Basisklasse wird um die Ereignis- bzw. Messpunktschnittstelle erweitert. Das direkte Ererben der Funktion von der implementierten OSIS-Basisklasse ist nicht möglich, da es so zu Problemen bei der Zuordnung der vom IDL-Compiler generierten Kommunikationsmethoden kommt. Folglich muss die Vererbung in die IDL-Schnittstellen verlagert werden. Da die Schnittstellen jedoch nur eine Beschreibung der Funktion ohne deren Algorithmen sind, ist es nicht möglich, die Funktion der OSIS-Basisklasse auf direktem Weg zu erben. Das Problem kann umgangen werden, indem in die Ereignis- bzw. Messpunktklasse eine nach außen nicht sichtbare Instanz der OSIS-Basisklasse eingebettet wird, an welche alle Aufrufe der OSIS-Basisklasse weitergeleitet werden (siehe Abbildung 43).

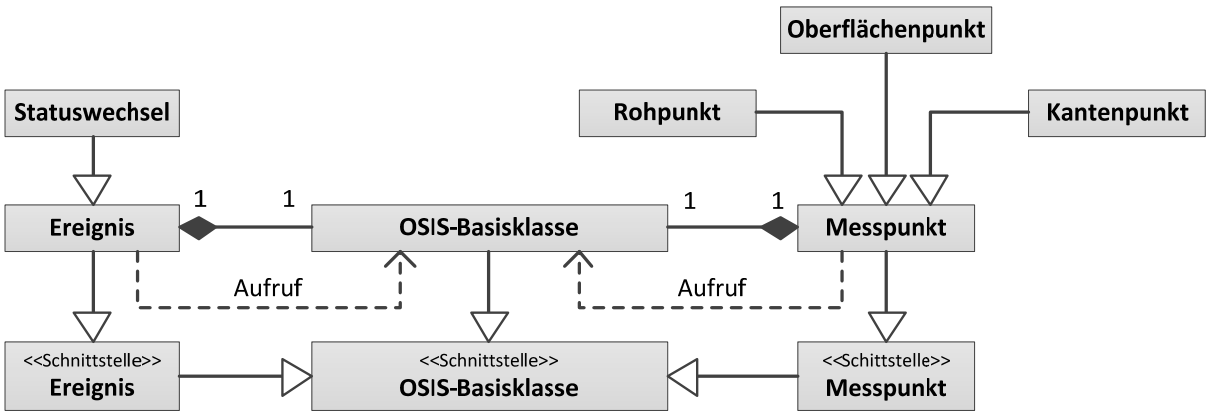


Abbildung 43: Aufrufdelegation an eine eingebettete Instanz der OSIS-Basisklasse

Neben der reinen OSIS-Funktionalität beinhaltet das Framework zusätzlich eine effiziente Fehlerbehandlung. Da OSIS-Fehler als spezielle Typen in I++DME integriert sind und auch somit von der zentralen NPMM-Steuerung behandelt werden müssen, ist die Fehlerbehandlung des verteilten Softwaresystems im OSIS-Adapter zentralisiert. Alle im OSIS-Rahmenwerk auftretenden Fehler werden an den OSIS-Proxy übertragen, welche entweder die Fehler selbst behandelt oder schwerwiegende Fehler an die NPMM-Steuerung weiterreicht.

Dieses Gerüst stellt eine Grundlage für die schnelle und OSIS-konforme Integration optischer Sensoren in ein I++DME Umfeld dar. Im nachfolgenden Kapitel wird auf dieser Grundlage die Steuerung für einen flächenhaft messenden Weißlichtinterferenzmikroskopsensor aufgebaut und damit für beliebige I++DME konforme Messsoftware anwendbar gemacht. Abbildung 44 illustriert die zeitliche Abfolge der Methodenaufrufe während einer Messung mit einem optischen Sensor in

Verbindung mit der OSIS-Erweiterung. Die hauptsächliche Kommunikation findet zwischen OSIS-Proxy und OSIS-Sensor über die Rechnergrenzen hinweg statt.

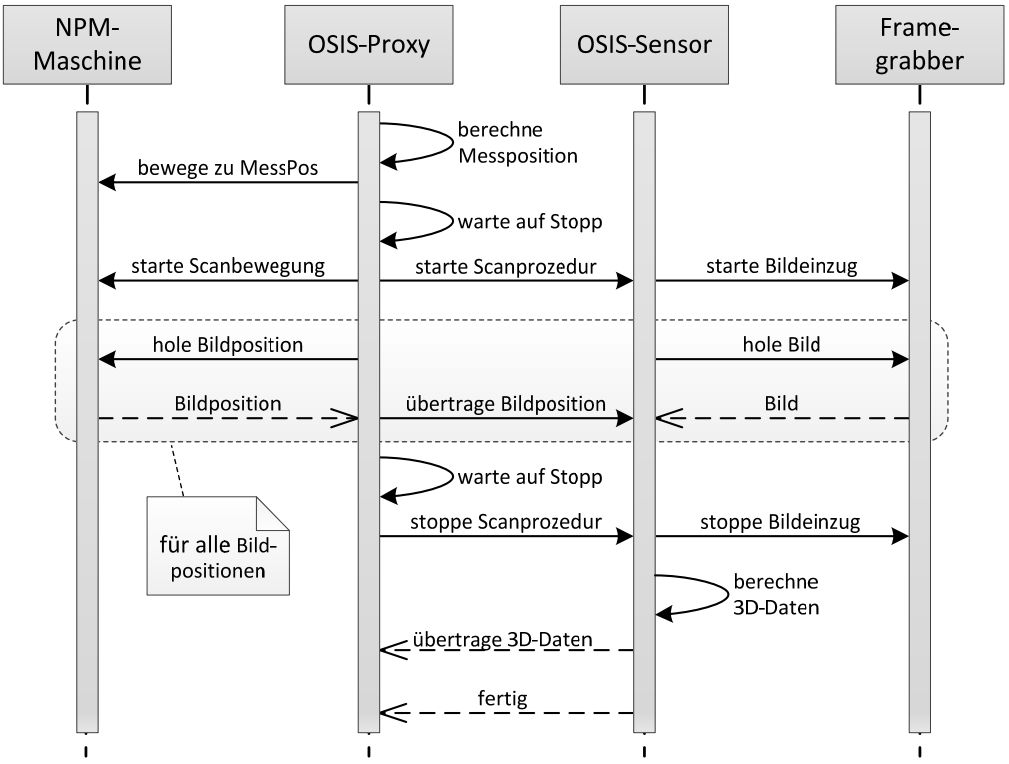


Abbildung 44: Interaktionsschema der OSIS Erweiterung

Nach Aufruf eines Messbefehls wird zunächst der Datenaufnahmeort ermittelt und die NPM-Maschine entsprechend positioniert. Nach Erreichen des Startpunktes wird der Bildeinzug gestartet und ein linearer Scan entlang der z-Achse ausgeführt. Während dieser Bewegung triggert die NPM-Maschine die Kamera und speichert parallel die Koordinaten der Triggerpositionen ab. Die Triggerpositionen werden von dem OSIS-Proxy ausgelesen und an den OSIS-Sensor übertragen. Parallel dazu werden die erfassten Bilddaten vom Framegrabber durch den OSIS-Sensor abgeholt. Nach Beendigung der Scanbewegung wird der Bildeinzug gestoppt und die Rohdaten werden vom OSIS-Sensor zu einem Höhenprofil der Messobjektoberfläche verarbeitet. Diese Höhenkarte wird anschließend an den OSIS-Proxy übertragen, welche daraus einen Antastpunkt extrahiert.

An diesem Punkt soll die Konzeption die OSIS-Erweiterung für den Demonstrator beendet werden. Die getroffenen Festlegungen bezüglich Architektur, Systemverteilung, Schnittstellen zu anderen Komponenten sowie Arbeitsweise und sinnvolle Implementierungsansätze wurden hinreichend erläutert. Im Folgenden soll dargelegt werden, wie die Sensorschnittstelle für optische Parallelsensoren eingebunden werden kann.

4.4.2.4 Proprietäre Schnittstelle für optische Parallelsensoren

Entwurf der Schnittstelle für optische Parallelsensoren

Da I++DME nur die Nutzung von Punktsensoren vorsieht und die Erweiterung nach OSIS diese Sensoren im weiteren Sinne emuliert, können die Vorteile optischer Arraysensoren nicht vollständig ausgeschöpft werden. Somit ist es erforderlich, den I++DME Befehlssatz proprietär zu erweitern. Mit der Schaffung einer weiteren Hardwareschnittstelle für optische Parallelsensoren wird eine zusätzliche Koppelstelle für optische Sensoren in die Steuerungssoftware integriert (siehe Abbildung 45).

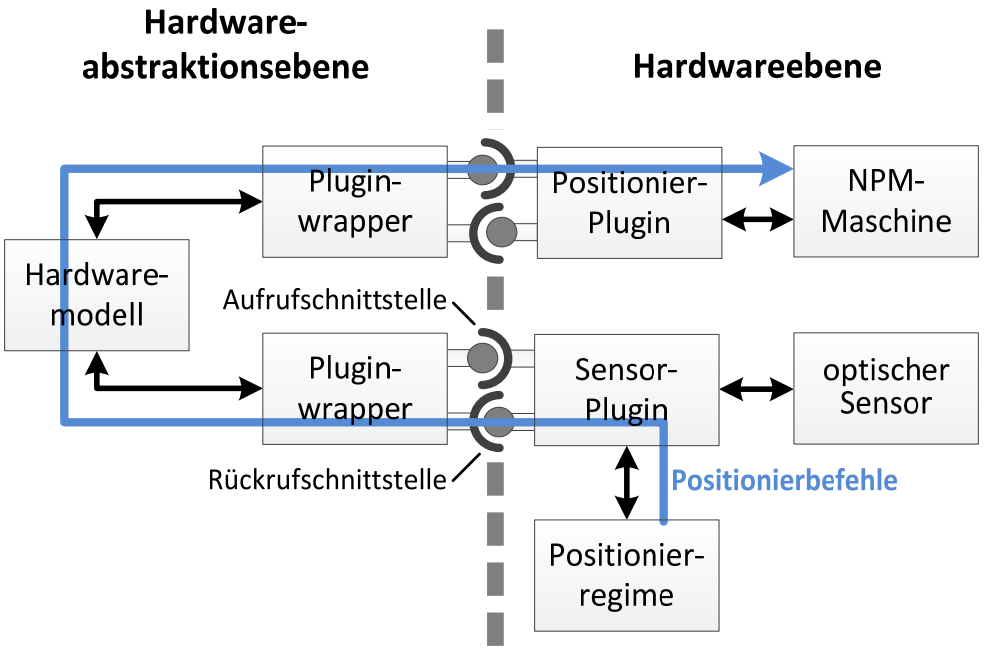


Abbildung 45: Integration der proprietären Schnittstelle für optische Sensoren

Um die Funktionalität der neuen Hardwareschnittstelle dem Anwender über die Nutzerschnittstelle zugänglich zu machen, wird der I++DME-Befehlssatz um zusätzliche Sensorparameter und Messbefehle speziell für optische Parallelsensoren ergänzt. Umfang und Inhalt der damit angefragten Messdaten soll entsprechend des bestehenden I++DME-Protokolls durch den Nutzer frei parametrierbar sein. Vorerst sollen zwei proprietäre Befehle für optische Sensoren implementiert werden. Ein Befehl, um eine einzelne Parallelmessung auszulösen und ein weiterer, um großflächige Bereiche per Stitching zu erfassen. Durch dieses Stitching ist es möglich, sehr große Objektbereiche im Rahmen einer Messung hoch aufgelöst zu erfassen. Jedoch ist dazu die Kalibrierung der Ausrichtung des Sensors relativ zum intrinsischen Koordinatensystem des Positioniersystems erforderlich, um diesen exakt positionieren und damit benachbarte Objektbereiche zusammen fügen zu können. Die Ausführungen zur Sensorkalibrierung folgen im Abschnitt 4.5. Aufgrund der a priori nicht spezifizierbaren Sensorhardware, müssen sowohl die Algorithmen für die Rohdatenverarbeitung als auch die Kalibrierverfahren vom Sensorplugin bereitgestellt werden. So bleiben die Mechanismen zur Erfassung und Verarbeitung der Rohdaten vollständig vor der Software der Hardwareabstraktionsebene verborgen, da ausschließlich die verarbeiteten 3D-Koordinaten an die Software der Hardwareabstraktionsebene übergeben werden.

Die optische Sensorschnittstelle basiert ebenso wie die übrigen Schnittstellen auf einem dynamischen Plugin-Mechanismus. Eine als Proxy dienende Wrapperklasse[3] ermöglicht zur Laufzeit den Austausch von Plugins, welche die Funktionalität des optischen Sensors bereitstellt. Der Befehlsumfang der Pluginschnittstelle umfasst Funktionen für die Kalibrierung des Sensors, zum Messen von einzelner Regionen oder großflächigen Bereichen sowie die Callbackfunktionen zur Übergabe von 3D-Punktdaten an die Steuerungssoftware. Durch die generische Gestaltung der Schnittstelle ist es möglich, beliebige Sensoren ohne einen expliziten Bezug zu den spezifischen Teilkomponenten zu verwenden. Sämtliche Routinen zur Kalibrierung, Datenerfassung und -verarbeitung werden, vergleichbar mit den Plugins der Maschinenschnittstelle, von einem Sensorplugin implementiert und von diesem zur Ausführung gebracht. Aufgrund der sehr verschiedenen Anforderungen optischer

[3] Ein Wrapper ist ein ummantelnder Softwareadapter. Wie bei mechanischen Bauteilen kann so ein inkompatibler oder mit einer allgemeinen Schnittstelle ausgestatteter Softwareteil konsistent in das Gesamtsoftwarekonstrukt eingebunden werden.

Messprinzipien muss die optische Sensorschnittstelle variabel genug gestaltet werden, um ein beliebiges Antastregime für die Rohdatenerfassung zu erlauben. Die dafür erforderlichen Aufrufe an das Positioniersystem erfolgen über entsprechende Callback-Schnittstellen der Sensorschnittstelle über das Hardwaremodell der NPMM-Steuerung hinweg. Nur so ist es möglich, optisch scannende Verfahren und Verfahren für Einzelaufnahmen sowie Punkt-, Linien- und Arraysensoren über eine gemeinsame Schnittstelle zu steuern.

Da die Sensoren während einer Messanwendung vom Anwender austauschbar sein sollen, ist es erforderlich, auch das Sensorplugin während einer I++DME Session wechseln zu können. Der Tausch wird beim Aufruf des dafür vorgesehenen I++DME Befehls durch die Softwarekomponenten der Hardwareabstraktionsebene vorgenommen. Die Identifikation eines Sensors als optischer Sensor erfolgt durch einen neuen, in die Werkzeugverwaltung des zentralen Hardwaremodells aufgenommenen Parameter, welcher die virtuelle Sensorrepräsentation als ein taktiles oder optisches Tastsystem ausweist.

4.4.3 Schnittstellen für zusätzliche metrologische Teilsysteme

Die Integration von Sensorwechseleinrichtung, Rotationsachsen und Bearbeitungswerkzeugen ist bisher unberücksichtigt geblieben, da diese Systeme für die volle Funktionalität einer iMERA-konformen NPM-Maschine jedoch erforderlich sind, wird ihre Integration in die Steuerungssoftware im folgenden konzeptioniert. Die folgenden Darstellungen gelten darüber hinaus aber auch für die Integration bisher nicht berücksichtigter Teilsysteme.

Als Ausgangspunkt für die Integration von zusätzlichen Hardwarekomponenten dient, wie bei den übrigen Systemen, die zentrale softwaretechnische Abbildung des Gesamtsystems. Das zentrale Hardwaremodell der Steuerungssoftware verfügt über eine abstrakte Repräsentation jedes einzelnen Teilsystems. Ist keine entsprechende Abbildung für die zu integrierende Hardware vorhanden, kann sie neu hinzugefügt werden. Diese Abbildung stellt der Steuerung zugleich den Zugang zur tatsächlichen Hardware bereit (siehe Abbildung 46), ohne jedoch bereits einen Bezug auf die konkrete Realisierung des Teilsystems zu nehmen. Die in der Hardwareabstraktionsebene befindliche Steuerungssoftware kennt nur die zum Teilsystem zugehörige Schnittstelle und sieht diese als Endknoten für den Datenaustausch an. Die dahinter befindliche Verarbeitung von Daten und Befehlen ist für die Steuerungssoftware transparent. Die eigentliche Umsetzung der Schnittstellen erfolgt, wie bei den bereits beschriebenen Schnittstellen über eine DLL-Schnittstelle, welcher jeweils ein Plugin-Wrapper zur Integration in die Steuerung und die Plugin-Adapter der Teilsysteme beigeordnet sind. Der Pluginwrapper stellt damit den Kommunikationsendknoten der objektorientierten Steuerungssoftware dar, welcher Aufrufe und Parameter über die Schnittstelle an den Adapter delegiert. Dieser muss die generischen Schnittstellenaufrufe in für das Teilsystem verständliche Aufrufsequenzen übertragen und seinerseits vom Teilsystem erhaltene Messdaten über die Rückruffunktionen der Schnittstelle an den Plugin-Wrapper und damit an die zentrale Steuerung übertragen.

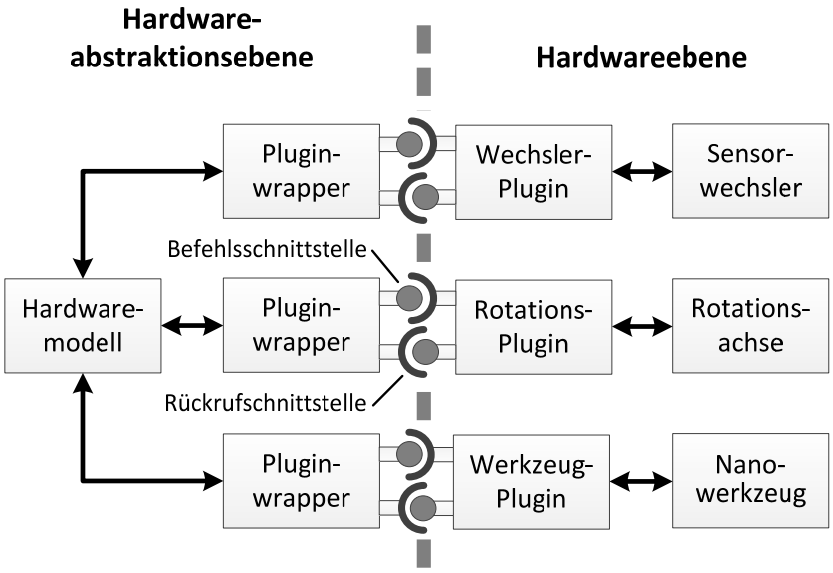


Abbildung 46: Konzept für die Integration von optionalen Schnittstellen

Die eigentliche Steuerung der zusätzlichen Komponenten erfolgt über den von der Anwenderschnittstelle bereitgestellten Standardbefehlssatz I++DME und muss, bedingt durch dessen Fokussierung auf die makroskopische KMT, auf unterschiedliche Weise umgesetzt werden, wenn nicht auf proprietäre Befehle zurückgegriffen werden soll. Die Integration der Sensorwechseleinrichtung gestaltet sich vergleichsweise einfach. Die vollständige Unterstützung des Multisensorkonzepts durch I++DME ermöglicht eine einfache Integration der Wechseleinrichtung in die Steuerungssoftware und die direkte Steuerung des Systems über I++DME-Befehle. Das zentrale I++DME Hardwaremodell repräsentiert alle verfügbaren Sensoren und Nanomanipulatoren in seiner internen Werkzeugdatenbank. Eine Zuordnung der Datenbankeinträge zur Einbauposition des Sensors in der Wechseleinrichtung ermöglicht den automatisierten Austausch. Im Rahmen der Abarbeitung eines Sensorwechsels kann zeitgleich mit der Zustandsaktualisierung des Hardwaremodells der Schnittstellenaufruf zum Tausch des Sensors erfolgen und es können gegebenenfalls erforderliche Sensorparameter zur Steuerung des Positioniersystems an die NPM-Maschine gesendet werden.

Die Abbildung von Rotationsachsen und Nano-Tools in den I++DME-Befehlssatz ist hingegen weniger einfach möglich. Da I++DME keine nativen Befehle für ein mehrachsiges Rotationssystem vorsieht und makroskopische Bearbeitungswerkzeuge in dem Bereich der Fertigungstechnik (im Speziellen der Computer Numerical Control, kurz CNC) zuzuordnen sind, fehlt eine Unterstützung dieser Systeme vollständig. Für die Steuerung von Nano-Tools käme nur in Frage, von der Steuerung nicht verwendete I++DME-Parameter, z. B. für nicht vorhandene Rotationsachsen des Sensors, zu instrumentalisieren oder proprietäre Befehle einzuführen. Im zweiten Fall sind diese Systeme nur mit entsprechend erweiterter Software steuerbar. Die Funktion der Rotationsachsen kann mit standardisierten I++DME-Clients zumindest teilweise hergestellt werden, da I++DME einachsige Rotationssysteme direkt unterstützt. Unter Umständen ist es möglich, die zweite Rotationsachse über einen von der Steuerung nicht verwendeten Parameter zugänglich zu machen oder den mehrachsigen Schwenktisch, wie die Antriebssteuerung der NPM-Maschine über einen I++DME-Pseudosensor zu integrieren. I++DME-Sensoren verfügen über bis zu drei Rotationsachsen, so dass alle Achsen des Schwenktisches direkt adressiert werden können. Auf diese Weise könnte das Fehlen der erforderlichen mehrachsigen Rotationsbefehle von I++DME ausgeglichen werden. Da diese Rotationssteuerung jedoch nicht konform zum eigentlichen I++DME-Bedienkonzept ist, könnten sich dadurch jedoch Auswirkungen z. B. auf die interne (Koordinaten-)Transformationskette bzw. eine eingeschränkte Parametrierbarkeit durch die verwendete I++DME-Clientsoftware ergeben.

Bei der zweckfremden Verwendung von I++DME-Parametern für die Steuerung von mehrachsigen Rotationssystemen und Nanotools ist es jedoch wahrscheinlich, dass es bei Nutzung von I++DME-Messsoftware mit hohem Automatisierungsgrad während der Messung zu Kollisionen kommen kann, da die Bahnplanungsalgorithmen der Messsoftware die Parameterdaten fehlinterpretieren würden.

4.4.4 Zwischenfazit

Das im Kapitel 3 vorgestellte Konzept zum Aufbau einer holistischen Steuerung für NPM-Maschinen ist damit hinreichend konkretisiert, um basierend auf NPM-Maschinen als Positioniersystem einen Demonstrator zu realisieren. Bei der Verfeinerung des Konzepts wurde die Art und Weise der Umsetzung aller Schnittstellen ausreichend dargelegt, um den eingangs gestellten Anforderungen an die zu entwerfende Software Rechnung zu tragen. Aufbauend auf diesem Konzept ist es einem Entwickler möglich, eine Hardware- und Steuerungssoftware unabhängige, frei skalierbare Steuerungssoftware für NPM-Maschinen umzusetzen, bei welcher die korrekte Verarbeitung von Messdaten durch die im Abschnitt 4.2.2 erfolgte analytische Verifikation der Schnittstellen gewährleistet ist.

4.5 Integration sensorspezifischer Kalibrier- und Korrekturverfahren

Im weiteren Verlauf des Kapitels wird ein Ansatz erarbeitet, um die noch ausstehenden Aspekte von hardwareunabhängiger Sensorkalibrierung und der nanometrologisch exakten Messdatenkorrektur konsistent in das Konzept der Steuerungssoftware zu integrieren. Dazu werden zunächst die von der Anwenderschnittstelle gegebenen Rahmenbedingungen erörtert und die Eigenschaften der konkreten Kalibrierverfahren vorgestellt.

4.5.1 Kalibrierverfahren für taktile Sensoren

Kalibrierung der Sensorauslenkung

Im Rahmen der Kalibrierung der Sensorauslenkung wird die Herstellung eines Bezugs zwischen Sensorsignal und der relativen Position des Sensors zu einem Messobjekt hergestellt. Das Ziel der Sensorkalibrierung bei taktilen Sensoren ist die Ermittlung einer Signal-Weg-Kurve und die Festlegung eines Arbeitspunktes anhand der ermittelten Daten. Der Arbeitspunkt legt den Zielwert der Antastkraft fest, mit welchem das Antasten im Rahmen einer späteren Messung erfolgen soll.

Tastkörperkalibrierung

Die Tastkörper nanometrologischer Mikro-Taster und AFM-Sonden weisen in vielen Fällen eine vergleichbare Ausdehnung wie die zu messenden Objektstrukturen auf und unterliegen ebenso wie die Tastkörper der makroskopischen KMT einer Formabweichung. Diese wird in der makroskopischen KMT meist vernachlässigt, gewinnt aber in der Nanometrologie eine besondere Bedeutung. Die Formabweichung ist in der Regel asymmetrisch und führt daher zu einer Verfälschung der Messdaten in Abhängigkeit von der Antastrichtung, welche mit den einfachen radiusbasierten Korrekturansätzen der makroskopischen KMT nicht korrigiert werden kann. Vielmehr müssen komplexe morphologische Verfahren (z. B. für dimensionelle Merkmale nach León [103], Dahlen [104] und Villarrubia [105] oder physikalische Eigenschaften nach Machleidt [106]) angewendet werden, um die Korrektur der systematischen Messfehler zu ermöglichen.

Dazu muss die tatsächliche Form des Tastkörpers exakt bestimmt und als Look-Up-Tabelle für die Tastkörperkompensation bereitgestellt werden. Sowohl der effektive Radius als auch die Look-Up-Tabelle können durch das Einmessen eines Kalibrierartefakts gewonnen werden. Soll lediglich der

effektive Radius ermittelt werden, genügt eine Präzisionskugel mit bekanntem Radius für diesen Vorgang. Für die Bestimmung der Oberflächen-Look-Up-Tabelle muss zusätzlich die Ausrichtung dieser Kugel bekannt sein. Die räumliche Ausrichtung kann gegebenenfalls mit einer Doppelkugel, d. h. mit zwei fest miteinander verbundenen Kugeln ermittelt werden. Im Rahmen der Tastkörperkalibrierung wird das Kalibrierobjekt vermessen (z. B. wie in Abbildung 47 dargestellt) und anhand der Messdaten und der bekannten Gestalt des Artefakts kann die Formgebung des Tastkörpers durch Differenzbildung ermittelt werden.

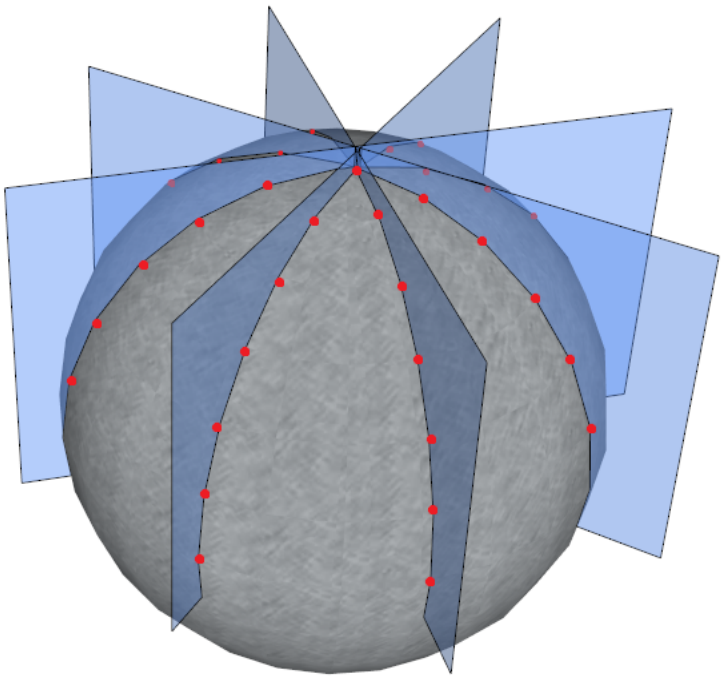


Abbildung 47: Anordnung von Scanebenen zur Bestimmung der Tastkörpergestalt an einem Kugelnormal

4.5.2 Kalibrierverfahren für optische Sensoren

In Anbetracht der vielfältigen optischen Messverfahren und ihrer jeweiligen, speziellen Zusammenhänge kann dieser Abschnitt nur die grundlegenden Zusammenhänge zur Bestimmung der intrinsischen sowie der extrinsischen Korrekturparameter wiederspiegeln, welche allgemeine Gültigkeit für die gesamte Klasse der optischen Sensoren besitzen. Etwaige weitere Zusammenhänge (z. B. bei Laserfokusmessungen die Ebenheit Fokusebene) müssen durch die Ergänzungen der im Folgenden dargestellten Kalibriermodelle und geeignete Verfahren zur Bestimmung der gesuchten Modellparameter ergänzt werden.

Intrinsische Kalibrierung

Vergleichbar mit der Kalibrierung der Sensorauslenkung ist die Ermittlung der intrinsischen Kameraparameter ein Verfahren zur exakten Beschreibung der Verzeichnung des Abbildungssystems, dass bei allen flächenhaft messenden optischen Parallelsensoren durchgeführt werden kann. Aufgrund von Unzulänglichkeiten des optischen Systems erfolgt die Abbildung des Messobjekts nicht ideal. Insbesondere in den Randbereichen wird das Abbild verzerrt dargestellt (siehe Abbildung 48). Die Abbildungsfehler äußern sich als tangentiale und radiale Verzeichnung. Die radiale Verzeichnung bewirkt eine polynomial gewichtete Verschiebung entlang des vom Hauptpunkt[4] ausgehenden Radius, wohingegen die tangentiale Verzeichnung zu einer Verzerrung entlang der Tangente führt. Die Ursache für die tangentialen Abbildungsfehler ist vornehmlich auf eine nicht orthogonale Ausrichtung

[4] Der Hauptpunkt entspricht bei einem optischen System dem Durchstoßpunkt der optischen Achse durch die Bildebene.

des Sensors zur Objektebene zurückzuführen, wohingegen radiale Abweichungen auf Unzulänglichkeiten des Abbildungssystems beruhen. In der Regel überlagern sich diese Abbildungsfehler und werden zusätzlich durch eine zufällige exzentrische Sensoranordnung richtungsabhängig verstärkt.

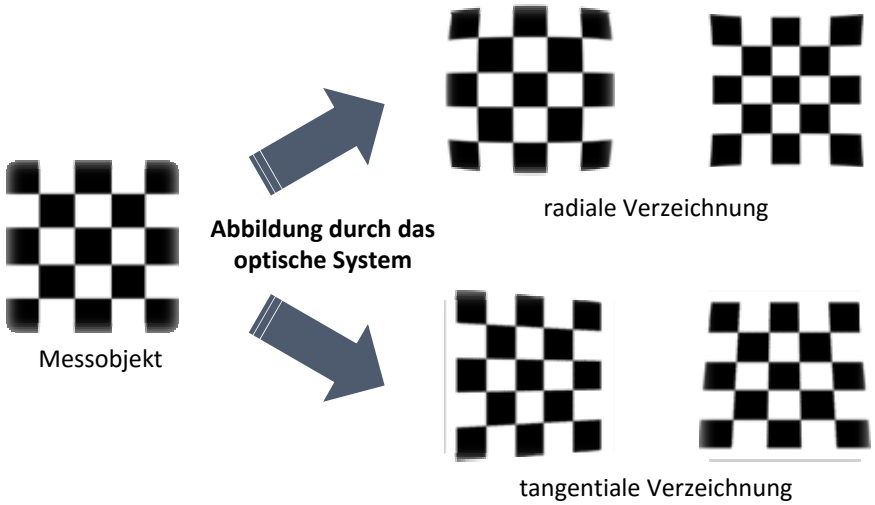


Abbildung 48: Schematische Darstellung der Verzeichnung an optischen Systemen

Da eine fehlerhafte Abbildung bei der Auswertung der Bilddaten statischen Messabweichungen führt, ist es zwingend erforderlich, die Abbildungsfehler rechnerisch zu korrigieren. Die mathematische Nachbildung der verzerrten Abbildung erfolgt durch das in Gleichung (4.4) wiedergegebene Modell [107] mit der Verzeichnungsfunktion $f(\vec{b})$.

$$\vec{b} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \cdot \vec{X} + t \tag{4.3}$$

$$\vec{x} = f\left([R|t] \cdot \vec{X}\right) \tag{4.4}$$

Die 3x4-Matrix $[R|t]$ des Lochkameramodells (4.3) bewirkt zunächst eine ideale Projektion der metrischen räumlichen Szene $\vec{X}$ auf die pixelbasierte Zwischenbildebene $\vec{b}$. Das Zwischenbild wird anschließend durch das polynomielle Verzeichnungsmodell (4.5) verformt. Die Modellparameter gliedern sich in zwei lineare Komponenten zur Beschreibung der radialsymmetrischen Verzeichnung (k_i) und der tangentialen Verzeichnung (p_i) auf. Die sich überlagernden Komponenten sind ein vergleichsweise einfacher Ansatz, die beschriebenen Verzeichnungseffekte mathematisch darzustellen. Sie erreichen jedoch bereits mit dem hier dargestellten Polynom dritten Grades eine hinreichende Nachbildungsgüte des Abbildungsfehlers.

$$f(\vec{\acute{x}}) = \begin{bmatrix} \acute{x} \cdot \dfrac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1 \acute{x}\acute{y} + p_2(2\acute{x}^2 + r^2) \\ \acute{y} \cdot \dfrac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1(2\acute{y}^2 + r^2) + 2p_2 \acute{x}\acute{y} \end{bmatrix} \tag{4.5}$$

mit

$$r^2 = \acute{x}^2 + \acute{y}^2$$

Im Rahmen der Verzeichnungskalibrierung werden die Parameter der Verzeichnungsfunktion $f(\vec{b})$ ermittelt. Durch die mathematische Umkehr des Verzeichnungsmodells kann der mit den Parametern

k_1 bis k_6 approximierte laterale Abbildungsfehler aus den Bilddaten weitgehend beseitigt und damit die Messabweichungen der optischen Sensoren insbesondere im Randbereich deutlich verringert werden.

Um die Modellparameter zuverlässig bestimmen zu können, sind hinreichend viele einander zugeordnete Bild-Objekt-Punktpaare für die Approximation erforderlich. Die Objektpunkte werden durch das Kalibriernormal vorgegeben und die zugehörigen Bildpunkte aus der Lage der Kalibriernormalstruktur im Sensorbild ermittelt. Als Kalibriernormal sind präzise gefertigte Schachbrett-Targets oder periodische Kreismuster zweckdienlich. Ihre Strukturen lassen sich durch ihre abrupten Helligkeitsgradienten im Abbild zuverlässig und mit hoher Präzision detektieren.

Extrinsische Kalibrierung

Ziel der extrinsischen Kalibrierung ist die Ermittlung einer Transformationsmatrix M (4.6), um die Projektionspunkte des pixelbasierten Kamerakoordinatensystems $\vec{x}$ in die metrischen Koordinaten des Positioniersystems $\vec{X}$ rückführen zu können.

$$\vec{X} = M \cdot \vec{x} \tag{4.6}$$

Ebenso, wie bei der Kalibrierung der intrinsischen Parameter wird auch für die Modellierung der extrinsischen Abbildung auf das Lochkameramodell (4.4) zurückgegriffen [108]. Äquivalent zur intrinsischen Kalibrierung werden mit einem Kalibriertarget einander zugeordnete Raum- und Bildpunkte erzeugt, auf deren Basis die Transformationsmatrix M approximiert wird. Dies sollte durch wiederholtes Positionieren des Normals und unter der Bezugnahme der vom Sensor detektierten Verschiebung auf die tatsächliche Verschiebung des Positioniersystems erfolgen, um die Rotation des Messobjekts exakt nachbilden zu können. Da dieses dynamische Verfahren iterativ ausgeführt werden kann, ist es möglich, damit eine Bootstrapping-Kalibrierung zu implementieren, welche eine initial geschätzte Transformationsmatrix M durch wiederholtes Neuberechnen präzisiert. So können die extrinsischen Parameter wesentlich exakter bestimmt werden, als es durch eine statische Kalibrierung möglich wäre.

Neben der direkten Überführung von optisch gemessenen Daten in das Weltkoordinatensystem ermöglicht eine exakt geschätzte Transformationsmatrix M das als Stitching bezeichnete, nahtlose Zusammenfassen benachbarter Bildbereiche zu einem Gesamtbild. Durch das gezielte Positionieren ist es damit möglich, mit einer minimalen Anzahl von Einzelaufnahmen eine hoch aufgelöste Gesamtansicht des kompletten Messvolumens zu erstellen.

Stereokalibrierung

Die Stereokalibrierung findet bei bi- oder multiokularen Sensorsystemen Anwendung, deren Zielstellung die dreidimensionale Erfassung einer räumlichen Szene ist. Generell wird versucht, die Sensoren dieser Systeme so anzuordnen, dass die optischen Achsen der einzelnen Abbildungssysteme entweder achsparallel liegen oder in einem gemeinsamen Punkt konvergieren. Die konvergente Anordnung hat dabei den Vorteil, dass der von den Sensoren beobachtete Szenenausschnitt maximiert und die spätere Suche nach homologen Punkten vereinfacht wird. In der Praxis ist es jedoch nahezu unmöglich eine exakte Konvergenz oder die genaue Achsparallelität herzustellen, wodurch die Sensoranordnung stets mehr oder weniger windschief ist. Durch eine nachträgliche affine Transformation der aufgezeichneten Bilddaten können die Szenenabbilder der Sensoren jedoch gezielt manipuliert werden, so dass eine virtuelle Ausrichtung entsprechend der gewünschten Kameraanordnung erreicht wird.

Die Voraussetzung für die Kalibrierung ist, dass ausreichend viele, einander zuordenbare Punkte in jedem Sensorabbild der Szene gefunden wurden. Um die Suche zu erleichtern, sollte für den Kalibrierprozess ein geeignetes Kalibriernormal verwendet werden. Zwar existieren auch Methoden und Ansätze, welche die erforderlichen Informationen direkt aus der Objektszene ermitteln, diese sind jedoch ungenauer und weniger stabil als die statische offline Kalibrierung.

$$\vec{x} = [R|t] \cdot \vec{X} \tag{4.7}$$

Über das Modell (4.7) werden die homologen Bildpunkte miteinander in Beziehung gesetzt, wodurch ein Gleichungssystem entsteht, dessen Lösung eine Schätzung der Transformationsparameter ermöglicht. Mit diesem können Bildpunkte eines Sensors in die intrinsischen Koordinaten der anderen Sensoren transformiert werden. Somit ist es auch problemlos möglich, eine zusätzliche affine Transformation für jeden Sensor festzulegen, womit alle Daten in ein gemeinsamen Raum abgebildet werden können. Dadurch wird die oben bereits angesprochene virtuelle Ausrichtung der Sensoren erreicht, wodurch bei einer Messung die Onlinesuche nach homologen Punkten deutlich effizienter gestaltet werden kann.

4.5.3 Kalibrierkonzept der I++DME-Schnittstelle

Für den Einsatz taktiler Sensoren im Rahmen nanometrologischer Anwendungen ist stets erforderlich, zumindest zwei Sensorparameter über entsprechende Kalibrierverfahren zu ermitteln. Zum Einen muss für die Antastregelung ein Arbeitspunkt auf der Signl-Auslenkungs-Kurve des Sensors festgelegt werden und zum Anderen ist es wegen der morphologischen Verfälschung der Messdaten durch die Gestalt des Sensortastkopfes erforderlich, dessen Gestalt für die nachträgliche Messdatenkorrektur exakt zu ermitteln. Allgemein sind den Möglichkeiten, Sensorparameter über die I++DME Schnittstelle hinweg messtechnisch zu bestimmen jedoch enge Grenzen gesetzt. Zum Einen abstrahiert das I++DME-Protokoll die Eigenschaften aller Sensoren auf wenige Basisparameter, mit denen lediglich eine Beschreibung der Sensorgeometrie mit geometrischen Formprimitiven möglich ist und zum Anderen ist nicht vorgesehen, Sensorrohdaten über die I++DME-Schnittstelle zu übertragen.

Die Messdatenkorrektur sollte von der NPMM-Steuerung durchgeführt werden, so dass an den Anwender ausschließlich korrigierte Messdaten übertragen werden. Ferner wird im I++DME-Standard explizit darauf hingewiesen, dass durch die NPMM-Steuerung Kalibrierverfahren bereitzustellen sind, welche vom Anwender über einen parameterlosen Schnittstellenaufruf gestartet werden können. Der Ablauf und die Umsetzung der Kalibrierverfahren obliegt somit vollständig dem Entwickler der Steuerungssoftware.

4.5.4 Integration von Messdatenkorrektur und Kalibrierverfahren für taktile Sensoren

Um den Grundgedanken des vollständig transparenten und dadurch hardwareunabhängigen Entwurfs der Steuerungssoftware beizubehalten, ist es erforderlich, die realisierungsspezifischen Kalibrier- und Messdatenkorrekturverfahren in die Hardwareebene zu transferieren (siehe Abbildung 49). Somit sind die Steuerungsplugins aller Positioniersysteme um entsprechende Algorithmen und Funktionen zu ergänzen. Im Sinne der Wiederverwendung ist es sinnvoll, die Kalibrier- und Messdatenkorrektur-algorithmen als gemeinsam genutzte Komponente in Form einer DLL zu implementieren, welche von allen Adaptern referenziert wird. Die Kalibrieralgorithmen können die Kommunikationspfade des Adapters verwenden, um das Positioniersystem zu steuern und die Rohdaten für die Parameterbestimmung in Empfang zu nehmen. Nach Abschluss der Kalibrierung werden der ermittelte Arbeitspunkt an die NPM-Maschine bzw. die Tastkörpergeometrie an die Softwarekomponente für die Messdatenkorrektur übertragen.

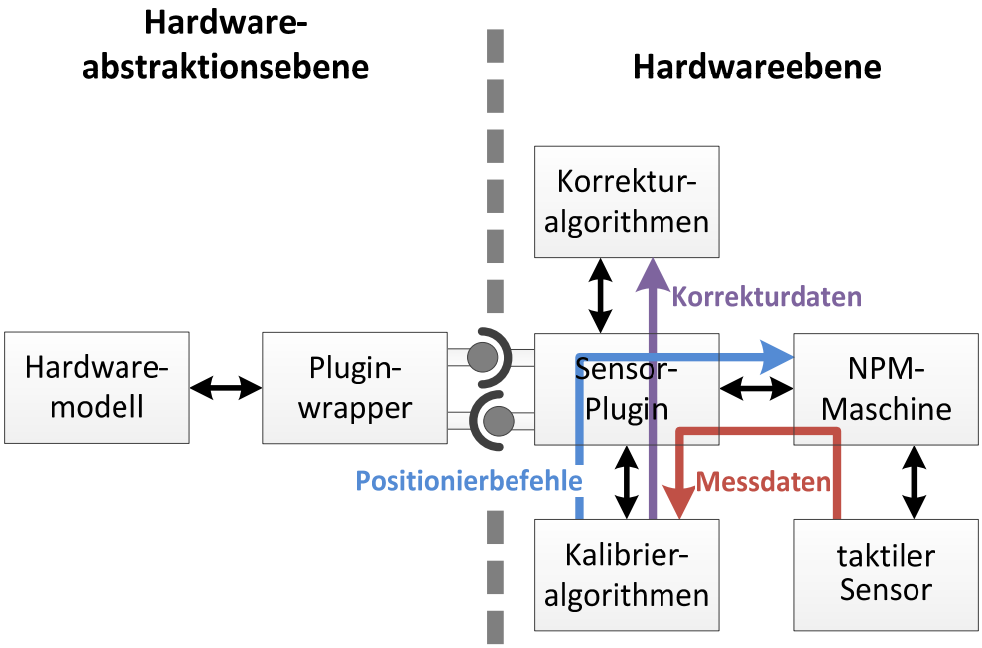


Abbildung 49: Integrationskonzept für Kalibrierverfahren und Messdatenkorrektur taktiler Sensoren

Da I++DME nur einen einzigen Befehl zum Starten aller Kalibrierroutinen definiert, müssen die beiden Verfahren entweder sequenziell und vollständig automatisiert ablaufen oder das Plugin kann für die Interaktion mit dem Anwender den nicht weiter spezifizierten I++DME Wartungskanal nutzen, um sich mit einer grafischen Nutzerschnittstelle für die Kalibrierung zu verbinden und eine manuelle Steuerung des Positioniersystems zu ermöglichen.

4.5.5 Integration der Kalibrierverfahren für optische Sensoren

Das Integrationskonzept für optische Kalibrierverfahren unterschiedet sich nicht maßgeblich von der Vorgehensweise bei taktilen Sensoren. Die Kalibrierverfahren werden auch hier an den Sensoradapter angegliedert (siehe Abbildung 50), wobei es zweckmäßig ist, Kalibrieralgorithmen mit allgemeinem Charakter als gemeinsam genutzte Komponente zu implementieren. Der maßgebliche Unterschied zum Vorgehen bei taktilen Sensoren ist, dass die Positionieraufrufe nicht direkt an das Positioniersystem gerichtet werden können. Um diesen Umstand zu beheben, bedienen sich die Kalibrieralgorithmen der in Abschnitt 4.4.2.4 dargelegten Vorgehensweise und greifen über die entsprechenden Rückruffunktionen der Sensorschnittstelle über die Hardwareabstraktionsebene indirekt auf das Positioniersystem zu. Nach Abschluss des Kalibrierprozesses werden die ermittelten Parameter an die Komponente für die Messdatenkorrektur übergeben, welche im Folgenden alle vom Sensor übertragenen Messdaten damit verarbeitet.

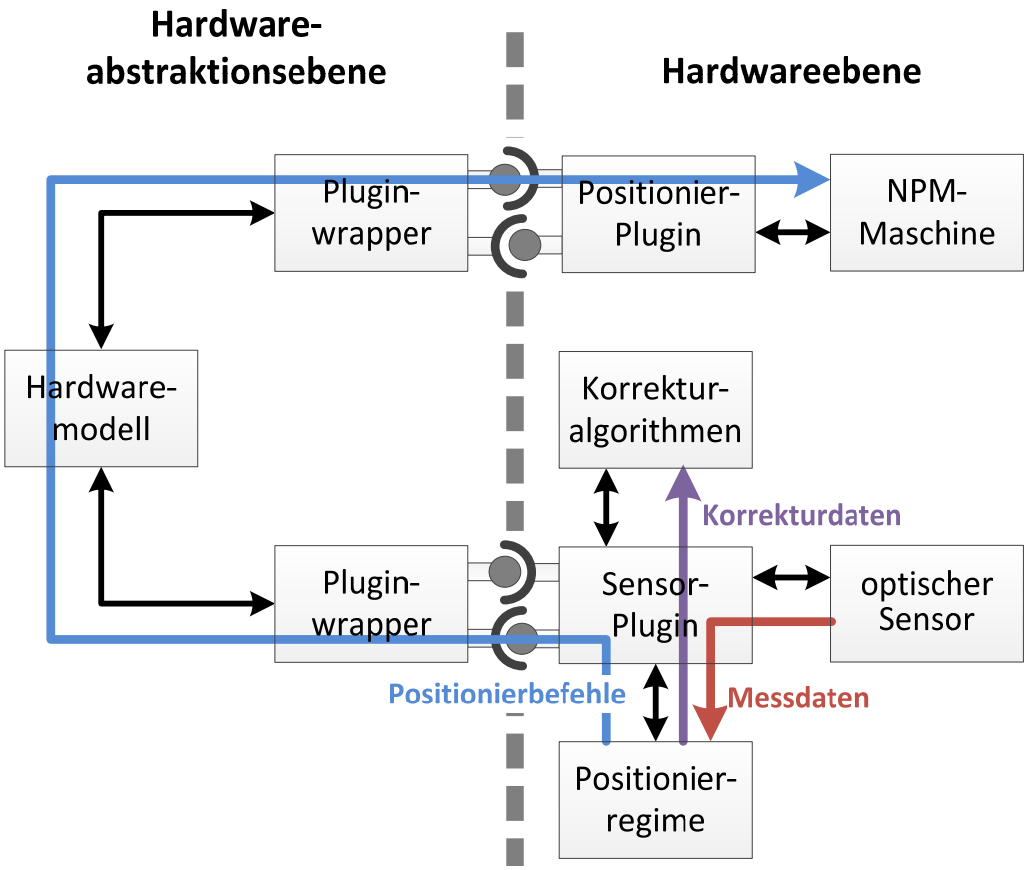


Abbildung 50: Integrationskonzept für Kalibrierverfahren optischer Sensoren

4.6 Fazit

In diesem Kapitel wurde das globale Konzept zum Entwurf einer Steuerungssoftware für NPM-Maschinen aus dem Kapitel 3 verfeinert. Aufbauend auf dem standardisierten I++DME-Modell wurde eine Architektur für die NPMM-Steuerung entwickelt, welche umfassende dynamische Schnittstellen für die hardwareunabhängige Integration metrologischer Teilsysteme bereitstellt. Mit Hilfe des vorgestellten Ansatzes ist es möglich, eine frei skalierbare NPM-Maschine zu entwickeln, welche beliebige Positioniersysteme, taktile und optische Sensoren, Nanotools, mehrere Rotationsachsen und eine Sensorwechseleinrichtung zu einem iMERA-konformen System kombiniert. Metrologische Aspekte wie die Kalibrierung der sehr unterschiedlichen Sensorsysteme stellen neben der Steuerung des Systems wichtige Funktionen für NPM-Maschinen dar und finden entsprechende Berücksichtigung im vorgestellten Entwurf. Die proprietären Kalibrierverfahren werden hardwarenah an die Sensoren gebunden, wodurch der eigentliche Kalibrierablauf vor der NPMM-Steuerung verborgen wird und damit über eine einheitliche Schnittstelle angesprochen werden kann.

Parallel zur Ausarbeitung des Integrationskonzepts wurde eine Demonstratorsoftware konzipiert, welche die maßgeblichen metrologischen Teilsysteme, wie verschiedene taktile und optische Sensoren und verschiedene NPM-Maschinen als Positioniersystem für den Anwender über die standardisierte Befehlsschnittstelle I++DME bereitstellt. Als Erweiterung zu I++DME wurde der Optical Sensor Interface Standard umgesetzt, wodurch es möglich ist, trotz der expliziten Ausrichtung von I++DME auf taktile Sensoren auch optische Sensorsysteme für metrologische Applikationen zu verwenden. Da OSIS jedoch naturgemäß nicht die volle Funktionalität optischer Parallelsensoren ausschöpfen kann, wurden speziell für diese Sensorklasse an I++DME angelehnte Erweiterungsbefehle geschaffen, welche den vollen Funktionsumfang optischer Sensoren über die Anwenderschnittstelle bereitstellen.

Kapitel 5 Modulare Integration optischer Sensorsysteme

Die im vorherigen Kapitel erarbeiteten Konzepte werden im Folgenden zu Softwarelösungen für die Integration optischer Sensoren in eine modulare NPM-Maschine ausgebaut. Eine wichtige Grundlage dafür bildet das flexible OSIS-Framework. Dieses Rahmenwerk ist ein generischer Softwarebaukasten, welcher, ergänzt durch sensorspezifische Rohdatenerfassungs- und Rohdatenverarbeitungskomponenten, die schnelle, effiziente und standardkonforme Entwicklung optischer Sensorsysteme ermöglicht.

Es soll Steuerungssoftware für drei optische Sensorsysteme entstehen, welche mit den Schnittstellen der NPMM-Steuerung kompatibel ist. Eines dieser Systeme basiert auf dem Messprinzip der Weißlichtinterferenzmikroskopie, erlaubt das schnelle, flächenhafte und sub-nanometergenaue Erfassen von Objektstrukturen und bildet durch diese Fähigkeiten eine optimale methodische Ergänzung zu anderen punktuell messenden optischen und taktilen Sensoren. Die beiden weiteren Sensorsysteme sollen, gestützt auf konventionelle Bildverarbeitungsalgorithmen, in vergleichsweise kurzer Messzeit 2D- bzw. 3D-Primärdaten mit geringer Auflösung bereitstellen. Die mit diesen Sensoren erzeugten Daten werden nicht direkt für die Lösung metrologischer Aufgaben verwendet, sondern stellen vielmehr eine Grundlage für die Planung und Optimierung des Messablaufs dar. Mit geeigneten optischen Verfahren ist das vollständige Erfassen eines ausgedehnten Messobjekts im Vergleich zu nanometrologischen Methoden sehr schnell (bis unter 1 s), aber nur grob aufgelöst (>100 µm) möglich. Diese Übersichtsdaten versetzen den Anwender jedoch in die Lage,

softwaregestützt Objektmerkmale für die spätere hochaufgelöste Messung zu identifizieren, wodurch die Planung und Optimierung der Messtrajektorie vereinfacht wird.

Im Fokus des Entwurfsprozesses der optischen Sensoren stehen neben der Strukturierung der jeweiligen Softwarekomponente die Entwicklung der für die Messdatenerzeugung erforderlichen Rohdatenverarbeitungsalgorithmen und die Eingliederung der Datenverarbeitungsalgorithmen in den vom OSIS-Framework bereitgestellten Kontext. Der generische Charakter des verwendeten Pluginkonzepts führt dazu, dass alle drei Sensorrealisierungen weitgehende konzeptionelle Parallelen aufweisen. Um Redundanzen zu vermeiden, wird die Implementierung des Weißlichtinterferenz-mikroskopsensors umfassend dargestellt und beim Entwurf der übrigen sensorischen Lösungen auf diese Darstellungen Bezug genommen.

5.1 Steuerungssoftware des Weißlichtinterferenzmikroskopsensors

5.1.1 Strukturierung der Sensorsoftware

Das zentrale Element der Sensorsoftware ist die Ablaufsteuerung (siehe Abbildung 51). Sie koordiniert die interne Datenverarbeitung mit den erforderlichen Aufrufen an die externen Komponenten des Positioniersystems. Die Positionieraufrufe werden bei Bedarf vom Positionsgenerator erzeugt und über die Callback-Funktionen der Pluginschnittstelle über die zentrale Hardwarebeschreibung der NPMM-Steuerung an das Positionierplugin übertragen. Die eigentliche Rohdatenerfassung erfolgt über einen Framegrabber, welcher durch die Ablaufsteuerung kontrolliert Rohdaten von der Sensorhardware anfordert und für die weitere Verarbeitung bereitstellt. Bevor die Rohdaten jedoch den WLI-Auswertungsalgorithmen bereitgestellt werden können, werden die Rohdaten mit den a-priori ermittelten Kalibrierparameter transformiert, um die optische Verzeichnung des Abbildungssystems zu korrigieren.

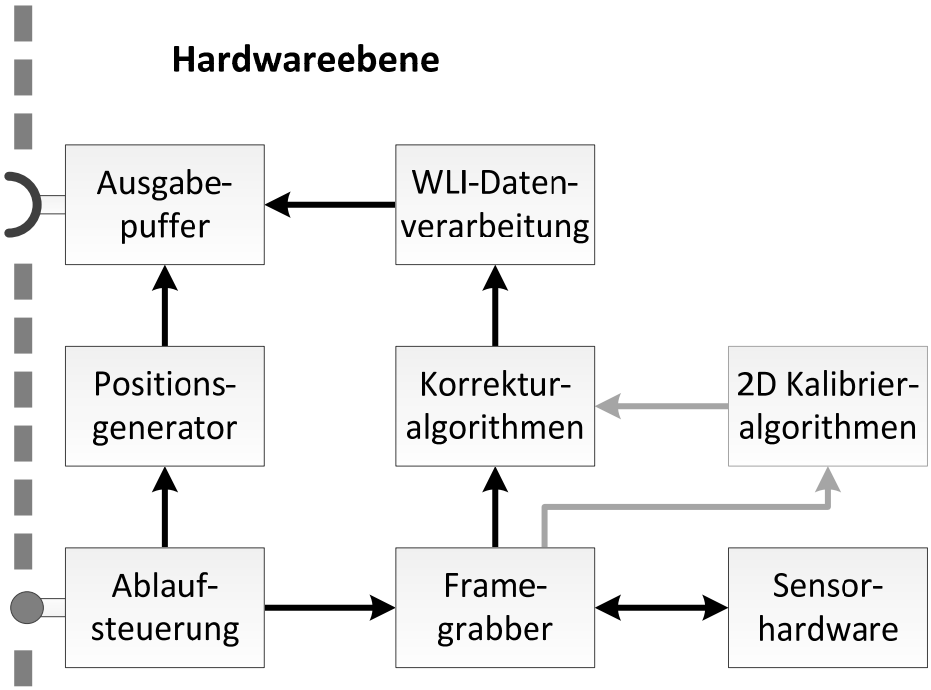


Abbildung 51: Konzept zum Aufbau der WLI-Sensor Software

Die funktionalen Zusammenhänge der affinen Transformation, die Kalibrierung sowie die Korrektur der Verzeichnung eines optischen Systems wurden in Abschnitt 4.5.2 dargelegt und können direkt in einen algorithmischen Ablauf überführt werden. Um die Entwicklung der Softwarekomponenten zu

beschleunigen, wurde auf die Bildverarbeitungsbibliothek Open Computer Vision (kurz openCV) [109] zurückgegriffen. Die openCV ist ein freies, nichtkommerzielles Softwareprojekt und kann unter den Bedingungen der Berkeley Software Distribution Lizenz verwendet werden. Die openCV umfasst alle grundlegenden Bildverarbeitungsalgorithmen, ist hoch optimiert und stellt umfangreiche Lösungsansätze für typische Anwendungsfälle der Bildverarbeitung bereit. Für die Entwicklung der optischen Sensoren wird im Rahmen dieser Arbeit auf die Kalibrier- und Korrekturlösungen der openCV zurückgegriffen. Die korrigierten Rohdaten werden anschließend an die Auswertungsalgorithmen übergeben. Diese Komponente wandelt die Rohdaten in metrologisch auswertbare Punktdaten um und legt diese 3D-Daten im Ausgabepuffer der Sensorsteuerung ab, wo sie für die Ausgabe an die zentrale NPMM-Steuerung bereit stehen.

5.1.2 Weißlichtinterferenzmikroskopie

5.1.2.1 Aufbau des WLI-Sensors

Den hardwareseitigen Aufbau des optischen Sensors ist in Abbildung 52 dargestellt. Die funktionsrelevanten Bauteile des optischen Sensors sind der Kollimator, ein Strahlteiler, das Mirau-Interferenzobjektiv [110] und der lichtempfindliche Kamerasensor. Der Strahlengang des seitlich eingekoppelten Lichts wird durch den Kollimator in einen parallelen Verlauf überführt und durch den Strahlteiler orthogonal in das kompakte koaxiale Interferenzobjektiv umgelenkt.

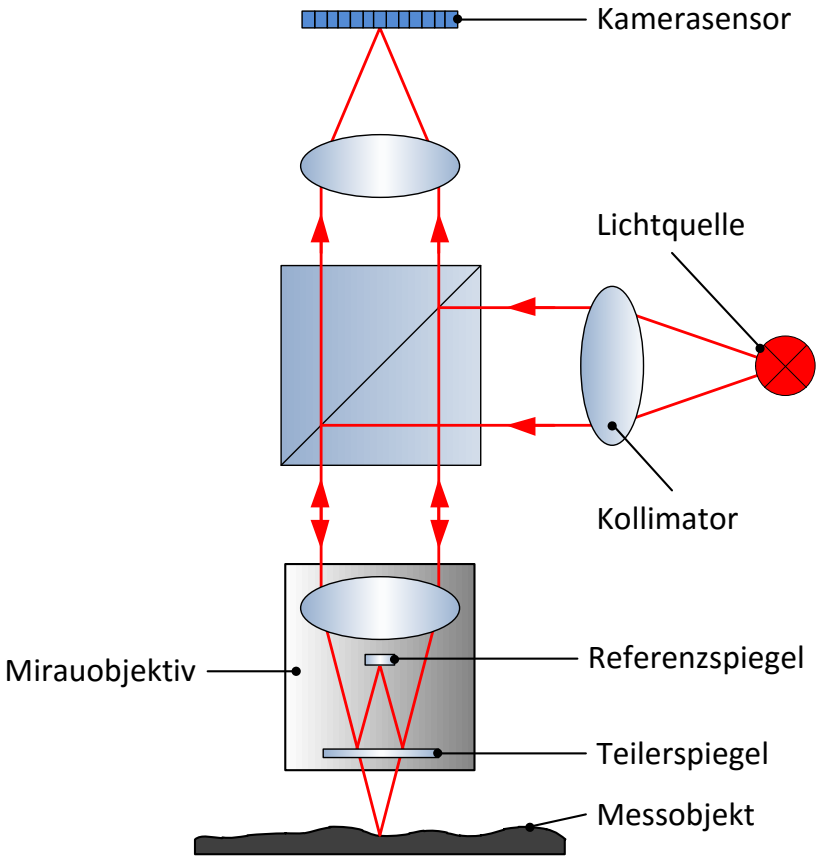


Abbildung 52: Aufbau des WLI-Sensors

Das Licht wird nach der Objektivlinse durch einen Teilerspiegel jeweils eine Hälfte zu einem Referenzspiegel reflektiert bzw. zum Messobjekt durchgelassen. Das vom Messobjekt reflektierte Licht wird partiell in das Interferenzobjektiv zurück reflektiert, wo es mit dem vom Referenzspiegel zurückgeworfenen Licht interferiert. Die durch Überlagerung erzeugte Interferenz wird als Intensität

vom Sensor, einem konventionellen Kamerasystem [111] erfasst, welcher sie in ein auswertbares Rohsignal für die nachfolgende rechnergestützte Datenverarbeitung umwandelt.

5.1.2.2 Messprinzip der Weißlichtinterferenzmikroskopie

Methodisch basiert die Weißlichtinterferenzmikroskopie auf der Interferenzbildung durch die Überlagerung von Licht einer breitbandigen Lichtquelle. Die Interferenzbildung kann als additive Überlagerung der Interferenzen einer Vielzahl untereinander nicht kohärenter monochromatischer Lichtquellen betrachtet werden, wobei sich die Gesamtintensität I_G einer Interferenzwelle nach Trittler [112] wie folgt bestimmt

$$I_G = I_1 + I_2 + 2\sqrt{I_1 I_2}\cos(\Delta\varphi) \quad \text{mit } \Delta\varphi = \varphi_2 - \varphi_1 \tag{5.1}$$

Bei der Superposition addieren sich die Intensitäten der sich überlagernden Wellen, wodurch es in Abhängigkeit von der Phasendifferenz Δφ zu konstruktiver oder destruktiver Überlagerung kommt.

$$I_G = \int_0^\infty I_1(\lambda) + I_2(\lambda) + 2\sqrt{I_1(\lambda)I_2(\lambda)}\cos(\Delta\varphi(\lambda))\,d\lambda \tag{5.2}$$

Aufgrund der unterschiedlichen Wellenlängen λ erfolgt die Bildung der jeweiligen Interferenzmaxima und -minima jedoch nicht mehr lokal korreliert (siehe Abbildung 53),

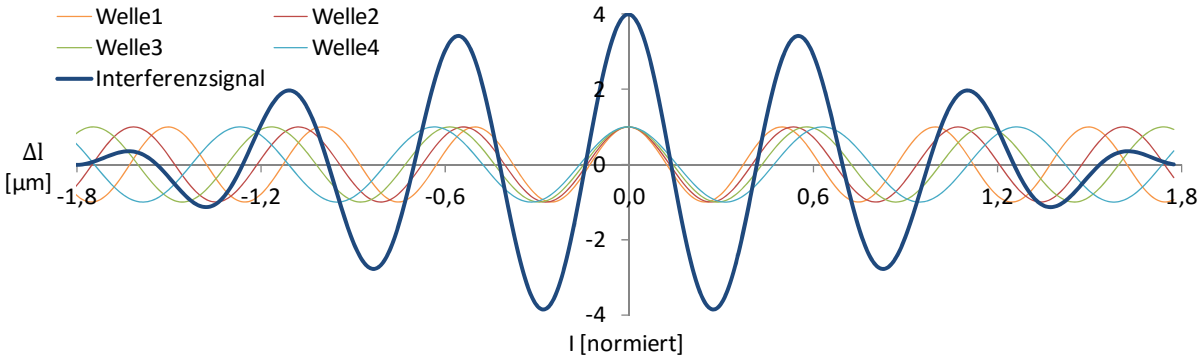


Abbildung 53: Überlagerung von Wellen und resultierendes Interferenzsignal

was in Abbildung 54 zusätzlich am Interferogramm einer Halogenlichtquelle verdeutlicht wird.

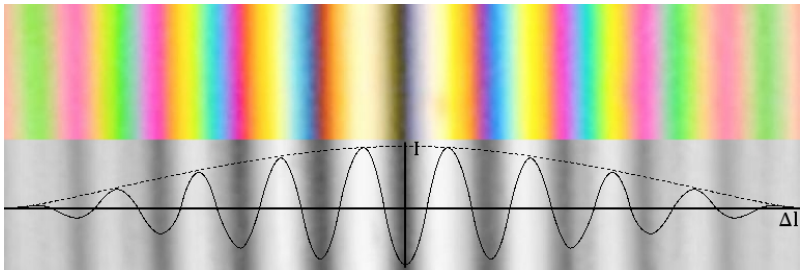


Abbildung 54: Interferogramm einer breitbandigen Halogenlichtquelle
(oben: farbige Darstellung, unten: Intensitätsverlauf mit der Einhüllenden)

Mit zunehmendem Wegunterschied Δl entstehen die Interferenzmuster der einzelnen Wellenlängen zueinander versetzt, wodurch das farbige Streifenmuster und die Intensitätsmodulation mit der dominanten Wellenlänge λ_0 des Lichtquellenspektrums entstehen. Da der Interferenzkontrast breitbandiger Lichtquellen mit wachsender Wegdifferenz bis zur Kohärenzlänge l_c zu Null abklingt, kann durch Bestimmung der Lage des Hüllenkurvenmaximums im Gegensatz zu monochromatischen

Lichtquellen die Bestimmung der Wegdifferenz auch über ein Vielfaches von $\Delta l \geq \lambda_0/4$ hinweg erfolgen.

Die Kohärenzlänge l_c ist ein Parameter des Lichtquellenspektrums und berechnet sich nach Seiffert [113] in Abhängigkeit der dominanten Wellenlänge λ_0 und der spektralen Bandbreite B, welche die Differenz zwischen oberer und unterer Grenzfrequenz der Lichtquelle darstellt, zu:

$$l_c = \frac{{\lambda_0}^2}{B} \tag{5.3}$$

Das Messverfahren der WLI-Mikroskopie basiert auf dem Vergleich von Referenz- und Messarmlänge (l_{ref} und l_{mess}) des sensorintrinsischen Interferometeraufbaus. Mit einem Positioniersystem wird die Messarmlänge etwa um die Distanz

$$l_{mess} \cong l_{ref} \tag{5.4}$$

in Schritten von

$$\Delta l_{mess} = \lambda_0/8 \tag{5.5}$$

variiert und an jeder Position

$$l_{mess} = k \cdot \Delta l_{mess} + l_{ref} \quad \text{mit } k \in \mathrm{Z} \tag{5.6}$$

die Intensität des Interferenzsignals aufgezeichnet. Nach Abschluss der Datenerfassung liegt das Interferogramm als diskreter Intensitätsverlauf des kontinuierlichen Interferenzverlaufs an den durch Gleichung 5.6 definierten Stützstellen im Datenverarbeitungssystem vor. Die Demodulation der abgetasteten Interferenzfunktion legt den Verlauf der Interferenzhüllkurve offen, auf deren Basis das Maximum der Hüllkurve und somit das Maximum des Interferenzkontrasts ermittelt werden kann [114]. Da der maximale Interferenzkontrast stets unter der Bedingung $l_{mess} = l_{ref}$ auftritt, ergibt sich die absolute Distanz d_{obj} zwischen angetastetem Messobjekt und der Mitte der Strahlteilereinheit zu

$$d_{obj} = i \cdot \Delta l_{mess} + l_{ref} \quad \text{mit } k_{max} \leq i \leq k_{min} \text{ und } i \in \mathrm{R} \tag{5.7}$$

Durch die Regression einer geeigneten, kontinuierlichen und konvexen Hüllfunktion an den diskreten Interferenzverlauf kann das (Interferenz-)Kontrastmaximum genauer als der Stützstellenabstand (siehe Gleichung 5.5) ermittelt werden. Die so erzielten Messergebnisse sind sehr hochauflösend und ermöglichen Reproduzierbarkeiten von wenigen Nanometern. Ein für optische Sensoren typischer Vorteil des WLI-Verfahrens ist die Parallelisierbarkeit von Messungen. Dazu wird anstatt des bisher betrachteten lichtempfindlichen Punktsensors ein Flächensensor in den Interferometeraufbau integriert. So ist im Rahmen einer Messung die nanometergenaue Antastung eine Vielzahl benachbarter Oberflächenpunkte möglich[5].

[5] Anstatt eines Flächensensors kann grundsätzlich auch ein Zeilensensor für die parallele Datenerfassung verwendet werden. Dieser wäre in der Lage, Messpunkte entlang einer Linie aufzuzeichnen. Aufgrund des WLI-Messregimes ist der Einsatz von Flächensensoren jedoch effizienter, weshalb Zeilensensoren für die WLI kaum eingesetzt werden.

5.1.2.3 Automatisierte Parameterschätzung

Der in Abschnitt 5.1.2.2 beschriebene Ansatz, die Distanz zum Messobjekt durch die Regression einer Hüllkurve zu ermitteln, ist einfach und dadurch anfällig für Störungen. Robuster und deutlich genauer ist es, das Interferogramm mit einem Matched-Filter zu falten und die Hüllkurvenregression und die Maximumsuche auf der Filterantwort auszuführen. Dazu muss der Filter jedoch entsprechend der spektralen Eigenschaften der Lichtquelle parametriert werden. Bisher wurden diese Parameter mittels langwieriger, manueller und ungenauer Heuristik geschätzt. Da es für die Verwendung der WLI-Algorithmen in einem optischen Sensor jedoch zwingend erforderlich ist, die Spektralparameter robust, schnell und automatisiert zu bestimmen, wurde das im Folgenden vorgestellte Analyseverfahren entworfen.

Der Algorithmus bedient sich der Tatsache, dass aufgezeichnete Interferogramme die gesuchten Informationen bereits enthalten. Im Detail werden für den Matched-Filter die Parameter *dominante Wellenlänge* λ_0 und die *Kohärenzlänge* l_c benötigt. Beide Eigenschaften können durch die Analyse von Interferogrammdaten im Fourierraum ermittelt werden. Aufgrund von Rauschen, Reflexionen und der Signalquantisierung durch die Rohdatenwandlung im Sensor sind die ermittelten Interferogramme jedoch gestört und es ist nicht hinreichend, die Parameter aus einem einzigen Datensatz zu schätzen. Somit müssen stabile Ergebnisse auf Basis einer großen Anzahl ($n \gg 100$) von Interferogrammen bestimmt werden. Das verwendete WLI-Sensorsystem ist jedoch in der Lage, Interferogramme ortsaufgelöst aufzuzeichnen, so dass eine hinreichend große Datenbasis für eine stabile Parameterbestimmung vorliegt. Die Festlegung der verwendeten Interferogramme I_i mit $i \in \{0..n\}$ erfolgt als gleichverteilte, deterministische Auswahl über alle vom Sensor aufgezeichneten Interferogramme.

Unter der Annahme einer temporär konstanten Bestrahlungsstärke entspricht jedes Interferogramm einem, von der Messobjektdistanz l_{mess} abhängigen Intensitätsverlauf $I(l_{mess})$. Der beobachtete Intensitätsverlauf ist nur abhängig vom Abstand z zwischen Messobjekt und Sensor. Die Aufzeichnung eines Interferogramms erfolgt an den diskreten Stützstellen z_j mit $j \in \{0..m\}$.

$$I_i(z) = I\left(x, y, z_j\right) \tag{5.8}$$

Das an einem Sensorpixel aufgezeichnete Interferogramm I_i kann aufgrund der angenommenen Konstanz des lateralen Ortes als Vektor wie folgt dargestellt werden

$$\vec{I_\iota} = \begin{Bmatrix} I(z_0) \\ I(z_1) \\ I(z_2) \\ \vdots \\ I(z_m) \end{Bmatrix} \tag{5.9}$$

Die Fouriertransformierte des diskreten Interferogramms entspricht dem Quadrat des einwirkenden Lichtquellenspektrum $\widetilde{I_\iota}$, da das Interferogramm eine Autokorrelationsfunktion repräsentiert.

$$\vec{I_\iota}(z) \circ\!\!-\!\!\bullet\, \tilde{I}_i^2(k) \tag{5.10}$$

Durch Approximation mit der Methode der kleinsten Fehlerquadrate werden aus dem Spektrum die dominante Wellenlänge λ_0 und die Kohärenzlänge l_c geschätzt. Als Schätzmodell $s\left(\widetilde{I_\iota}\right)$ für das Spektrum dient eine, um die Amplitude a skalierbare und in der Intensität um einen Offset o verschiebbare Gauß'sche Normalverteilung, welche sich als Näherung für Leuchtdioden und Plancksche Strahlungsquellen sehr gut eignet.

$$s_i(\widetilde{I_i}) = \frac{a_i}{\sigma_i\sqrt{2\pi}} e^{-\frac{\sqrt{\widetilde{I_i}(k)}-\lambda_{0i}}{2\sigma_i}} - o_i \tag{5.11}$$

Zur Bestimmung der Kohärenzlänge l_c ist zunächst die Bandbreite B des Spektralmodells zu ermitteln.

$$B_i = 2\sqrt{2\ln 2}\,\sigma_i \tag{5.12}$$

Die Kohärenzlänge l_c für jedes Interferogramm I_i kann mit den Gleichungen (5.3) und (5.12) bestimmt werden

Diese Prozedur wird für alle n Interferogramme durchgeführt und die ermittelten Parameter werden jeweils in einem Merkmalsvektor $\overrightarrow{m_i}$ abgelegt. Um dem Rauschen zu begegnen, wird aus der Menge aller ermittelten Parametersätze ein optimales Paar ausgewählt. Eine Möglichkeit wäre, durch einfache Mittelwertbildung den Schwerpunkt der Parameterverteilung zu ermitteln und als optimalen Parametersatz anzusehen. Jedoch geht so der Bezug zu den tatsächlich ermittelten Daten verloren und Ausreißer erhalten ein höheres Gewicht. Deswegen wird der am nächsten zum Schwerpunkt liegende Parametersatz ausgewählt, um den Messdatenbezug zu erhalten. Dafür wird zunächst der Schwerpunkt $\overline{m}$ der Parameterwolke bestimmt

$$\overline{m} = \frac{1}{n}\sum_{i=0}^{n} \overrightarrow{m_i} \tag{5.13}$$

mit $\overrightarrow{m_i}$ als Merkmalsvektor des Interferogramms I_i, welcher daran ermittelten spektralen Parameter zusammenfasst.

$$\overrightarrow{m_i} = \begin{Bmatrix} l_{c\,i} \\ \lambda_{o\,i} \end{Bmatrix} \tag{5.14}$$

Da λ_0 und l_c unterschiedlich streuen ($\sigma_{\lambda_0} \neq \sigma_{l_c}$), muss der Abstand zum Schwerpunkt über die Mahalanobis-Distanz ermittelt werden. Im Rahmen der Distanzbestimmung wird der Parameterraum durch eine Normierung entlang der Eigenvektoren skaliert. Durch diese Normierung ist sichergestellt, dass beide Parameter gleich gewichtet in das Distanzmaß eingehen und eine breit streuende Größe die kompaktere Eigenschaft nicht überlagert.

Der Abstand des Merkmalsvektors $\overrightarrow{m_i}$ zum Schwerpunkt $\overline{m}$ wird wie folgt bestimmt

$$d_i(\overrightarrow{m_i}) = \sqrt{(\overrightarrow{m_i} - \overline{m})^T C^{-1} (\overrightarrow{m_i} - \overline{m})} \tag{5.15}$$

mit

$$C = \begin{pmatrix} \mathrm{cov}(\overrightarrow{p_i}, \overrightarrow{p_i}) & \mathrm{cov}(\overrightarrow{p_i}, \overrightarrow{p}) \\ \mathrm{cov}(\overrightarrow{p}, \overrightarrow{p_i}) & \mathrm{cov}(\overrightarrow{p}, \overrightarrow{p}) \end{pmatrix} \tag{5.16}$$

und

$$\mathrm{cov}(\vec{x}, \vec{y}) = \frac{1}{n}\sum_{i,j=0}^{n} (\vec{x} - \bar{x})(\vec{y} - \bar{y}) \tag{5.17}$$

Die Distanz d_i ist stets positiv und das optimale Parameterpaar

$$\overrightarrow{m_{opt}} = \overrightarrow{m_i} \tag{5.18}$$

kann unter der Bedingung

$$\overrightarrow{m_i} = \min_i(d_i) \tag{5.19}$$

ausgewählt werden.

Nach Abschluss der Parameterschätzung repräsentiert das Parameterpaar $\overrightarrow{m_{opt}}$ die modellhafte Beschreibung des Lichtquellenspektrums durch die dominante Wellenlänge λ_0 und die Kohärenzlänge l_c. Aufgrund der Parameterselektion mittels Mahalanobis-Distanz bleibt zudem der Bezug zu den Rohdaten erhalten, wodurch es möglich ist, die Parameter direkt einem eingangsseitig vorliegenden Interferogramm und damit einem Sensorpixel zuzuordnen.

Zeitgleich wurde von Date [115] ein vergleichbarer Algorithmus entwickelt, welcher im Gegensatz zum oben dargelegten Verfahren den Hauptteil der Berechnung im Ortsbereich durchführt, wodurch davon auszugehen ist, dass die Verarbeitungsdauer im Vergleich zur oben dargestellten Berechnung im Fourierraum annähernd verdoppelt wird. Die Gegenüberstellung der Leistungsfähigkeit beider Ansätze erfolgt im Rahmen der Systemvalidierung in Kapitel 6.4. Dort werden die Verarbeitungsdauer und die Schätzgüte für die spektralen Parameter an beiden Algorithmen untersucht und im Bezug auf die unterschiedliche Implementierungen erörtert.

5.1.3 Entwurf der WLI-Datenverarbeitungskomponente

Eine Beschreibung der im Rahmen der Arbeit adaptierten WLI-Verarbeitungsverfahren wird durch Kapusi in [62], [19] und [116] sowie von Machleidt in [117], [118] und [119] gegeben. Darüber hinaus können weitere Darstellungen von Verfahren zur Verarbeitung von Interferenzserien im Allgemeinen den Arbeiten von Trittler [112], Seiffert [113] und Hißmann [120] entnommen werden.

Die vorhandenen WLI-Verarbeitungsalgorithmen erforderten für die Integration in den modularen Entwurf eines OSIS-konformen optischen Sensors jedoch ein Überarbeitung, in deren Rahmen umfangreiche Anpassungen durchgeführt wurden. Die ursprüngliche Implementierung der WLI-Verarbeitung wurde als geradlinige Pipeline entsprechend der Darstellung in Abbildung 55 entworfen. Da die Verarbeitungsalgorithmen nicht im Rahmen dieser Arbeit entstanden, wird ihre Funktionsweise nur knapp erläutert und die einzelnen Module im Folgenden als opake Funktionsblöcke betrachtet. Hell hervorgehobene Blöcke stellen Funktionskomponenten dar und dunkle Blöcke repräsentieren Daten bzw. Datencontainer. Jeder Funktionsblock umfasst eine Menge von Klassen, welche die Algorithmen für die Lösung der jeweiligen Aufgabe bereitstellen.

Die Aufgabe der *Vorverarbeitung* ist es, die gemessenen Interferenzrohdaten entgegen zu nehmen und die Datenmenge durch intelligentes Einfügen in den *Interferenzbildstapel* zu reduzieren. Der *Interferenzbildstapel* ist ein Datencontainer, welcher zur pixelweisen Speicherung der von der *Vorverarbeitung* detektierten Interferogramme verwendet wird. Nachdem alle Rohdaten an die Pipeline übergeben wurden, erfolgt die Weiterverarbeitung der Daten durch die *Hauptverarbeitung*. Entsprechend ihrer Parametrierung nutzt diese ein Schätzverfahren, um aus dem *Interferenzbildstapel* einen Höhenwert für jedes Pixel zu berechnen und in der *Höhenkarte* abzulegen. Hat die *Hauptverarbeitung* ihre Arbeit beendet wird der *Interferenzbildstapel* gelöscht und die eigentliche Auswertung der WLI-Daten ist beendet. Eine Weiterverarbeitung durch die *Nachverarbeitung* ist optional und dient dazu, benachbarte Messfelder zu einer Gesamtrepräsentation zu kombinieren. Etwaig auftretende Fehlermeldungen werden von allen Funktionsblöcken an den *Fehlerspeicher* übergeben. Der Anwender ist so in der Lage, die Ursache eines Fehlers zu ermitteln.

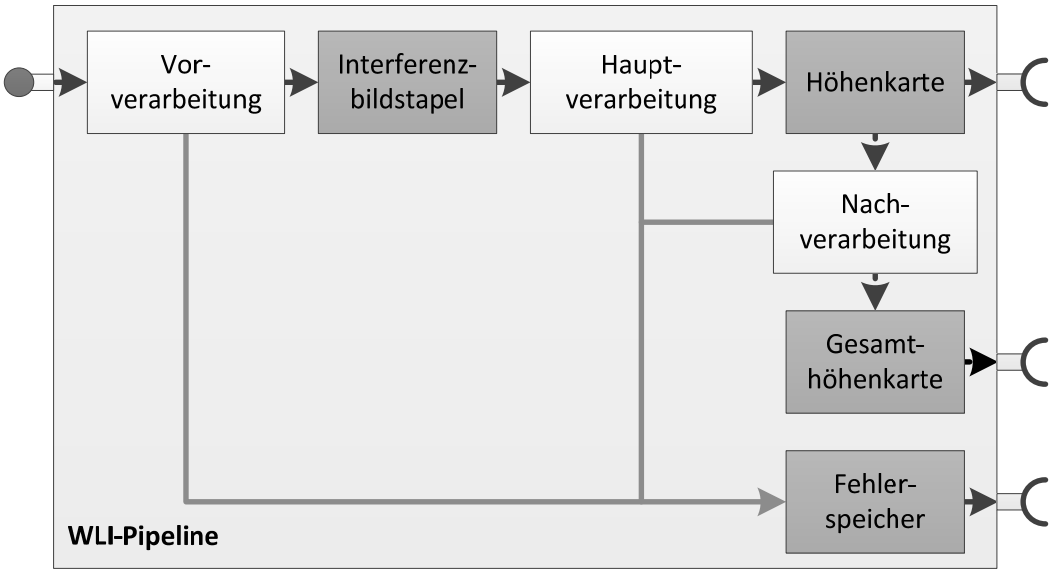


Abbildung 55: Ursprünglicher Aufbau der WLI-Datenverarbeitung als Pipeline

Der Aufbau und die Struktur der bestehenden WLI-Pipeline sind für die Verarbeitung von Interferenzdaten grundlegend geeignet. Es ist jedoch nachteilig, dass die Pipeline nur im Ganzen initialisiert und abgearbeitet werden kann. Darüber hinaus können einmalig aufgezeichnete Daten nicht durch mehrfachen Durchlauf der Hauptverarbeitung mit unterschiedlichen Einstellungen verarbeitet werden, wodurch die Auswirkungen dieser Veränderungen auf das Ergebnis nicht untersucht werden kann. Ferner existiert für die WLI-Pipeline keine Unterstützung für eine automatisierte Parameterwahl. Da die Abstimmung der Lichtquellenparameter aber entscheidend für die Qualität der Ergebnisse ist, mussten bisher langwierige Versuche mit wiederholtem Abarbeiten der kompletten Pipeline durchgeführt werden, um einen günstigen Parametersatz heuristisch zu ermitteln. Im Zuge der beabsichtigten Kapselung zu einer Funktionsbibliothek sollen die verschiedenen Unzulänglichkeiten behoben werden. Die zu entwickelnde Bibliothek soll über eine größtmögliche Portabilität verfügen, ein mehrmaliges Abarbeiten von Rohdaten erlauben, die Lichtquellenparameter ermitteln und für schnelle Untersuchungen eine Option bieten, selektiv einzelne Pixel zu verarbeiten und neben dem eigentlichen Ergebnis für umfangreiche Analysen auch Zwischenergebnisse bereitstellen.

Um diese Anforderungen zu erfüllen, wurde die Pipelinearchitektur aufgebrochen, ihre objektorientierten Module angepasst und mit ergänzenden Funktionsgruppen und Datencontainern entsprechend Abbildung 56 neu verknüpft und zu einer Funktionsbibliothek zusammengefasst. Da keine allgemeine Richtlinie für die Darstellung von objektorientierten Instanzen im Hauptspeicher existiert, ist es nicht möglich, die Funktionsbibliothek mit einer objektorientierten Schnittstelle zu versehen. Klasseninstanzen werden im Hauptspeicher proprietären Regeln folgend hinterlegt, was die Nutzung eines mit Compiler A erzeugten Objektes durch den Compiler B verwehrt. Exemplarisch wurde die Interoperabilität exportierter Klassen zwischen den Microsoft C++ Compilern der Version 6, 9 und 10 und dem Borland C++ Compiler 5.11 untersucht. Es war bei keiner Compilerkombination möglich, exportierte Klassen fehlerfrei einzubinden. Somit ist festzuhalten, dass die Interaktion des Maschinencodes verschiedener Compiler bei der Nutzung einer objektorientierten Schnittstelle nicht möglich ist, wodurch für eine portable WLI-Bibliothek auf eine konventionelle ANSI-C Schnittstelle zurückgegriffen werden muss. Dadurch wird die Integration der Bibliothek in das objektorientierte Gesamtkonzept minimal aufwendiger, bietet jedoch den Vorteil einer breiteren Einsatzmöglichkeit im Rahmen anderer Applikationen.

Unter der Beschränkung auf bestimmte Datentypen und eine gemeinsame Hardwarearchitektur kann dennoch eine hinreichende Portabilität erreicht werden. Wird der Maschinencode nur zwischen Systemen mit einheitlicher Prozessorarchitektur portiert, gewährleistet die Beschränkung auf grundlegende Datentypen und einfache Datenstrukturen, unter Ausschluss der Objektorientierung, die Interoperabilität. Da die Basisdatentypen sich auf die Datenbreite der Prozessorregister beziehen, stellt die einheitliche Prozessorarchitektur die Schnittstelle auf Maschinencodeebene dar. Eine hinreichend standardisierte Untermenge grundlegender Datentypen und -strukturen wird zum Beispiel durch den Standard ANSI-C [121] bereitgestellt. ANSI-C verwendet keine komplexen Datenkonzepte, weshalb keine Probleme aufgrund unterschiedlicher Datenrepräsentation im Hauptspeicher auftreten. Somit ist eine ANSI-C-konforme Schnittstelle ein probates Mittel, um eine hinreichende Portabilität für Funktionsbibliotheken zu gewährleisten.

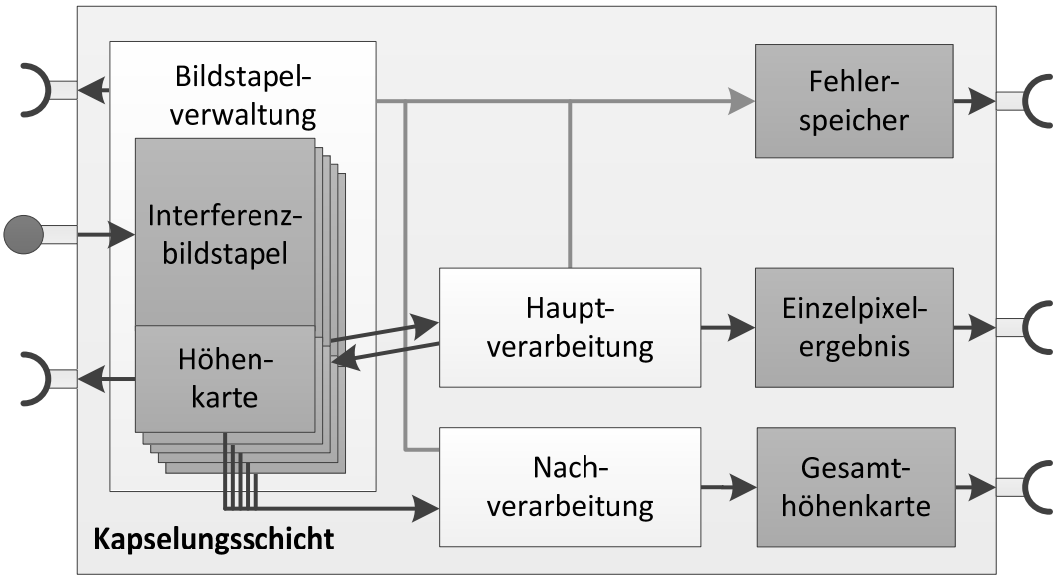


Abbildung 56: Aufbau der Abstraktionsschicht der WLI-Funktionsbibliothek

Die wesentlichste Änderung neben der Kapselung ist die Möglichkeit, mehrere *Interferenzbildstapel* parallel nutzen zu können. Es wurde die erweiterte *Bildstapelverwaltung* geschaffen, welche funktionell die *Vorverarbeitung* ersetzt und die *Interferenzbildstapel* mit jeweils einer angekoppelten *Höhenkarte* verwaltet. Der zuvor fest in der *Hauptverarbeitung* integrierte *Interferenzbildstapel* wurde herausgelöst und durch einen Verweismechanismus auf ein Element der *Bildstapelverwaltung* ersetzt. Mit Ausführung der *Hauptverarbeitung* wird der aktuell adressierte *Interferenzbildstapel* verarbeitet und die Ergebnisse in einer dem Bildstapel beigeordneten *Höhenkarte* abgelegt. Die Daten des *Interferenzbildstapels* werden nicht mehr gelöscht und können ohne weiteres mit anderen Parametern erneut verarbeitet werden. Das optionale Stitching durch die *Nachverarbeitung* wird auch weiterhin unterstützt. Dazu kombiniert die *Nachverarbeitung* alle gültigen *Höhenkarten* der *Bildstapelverwaltung* zu einer *Gesamthöhenkarte*. Die Schätzung der spektralen Lichtquellenparameter kann für einen im Vorfeld aufgezeichneten Interferenzbildstapel als Schnittstelle bereitgestellt werden. Für den Fall, dass nur bestimmte Bildbereiche analysiert werden müssen, wurde zusätzlich die Möglichkeit eingeführt, eine boolesche Maske über die Rohdaten zu setzen. Damit können beliebige Pixel von der Verarbeitung ausgeschlossen und so die Datenverarbeitung beschleunigt werden.

5.1.4 Integration der WLI-Komponenten in den OSIS-Kontext

Der Aufbau der Ansteuerungssoftware folgt den im vorigen Kapitel vorgestellten Konzepten und wird durch Sparrer in [122] und [123] eingehend beschrieben. Als Interaktionsplattform wird das OSIS-

Framework verwendet, dessen Komponenten für die Interaktion und Sensoransteuerung in zwei Verarbeitungselemente verteilt werden. Für die Ansteuerung der Sensorhardware und die Auswertung der gewonnenen Rohdaten sind weitere Komponenten erforderlich, welche das OSIS-Grundgerüst mit der eigentlichen Funktion des Sensors auskleiden. Die integrierten OSIS-Komponenten sind mit ihren Erweiterungen in Abbildung 57 veranschaulicht.

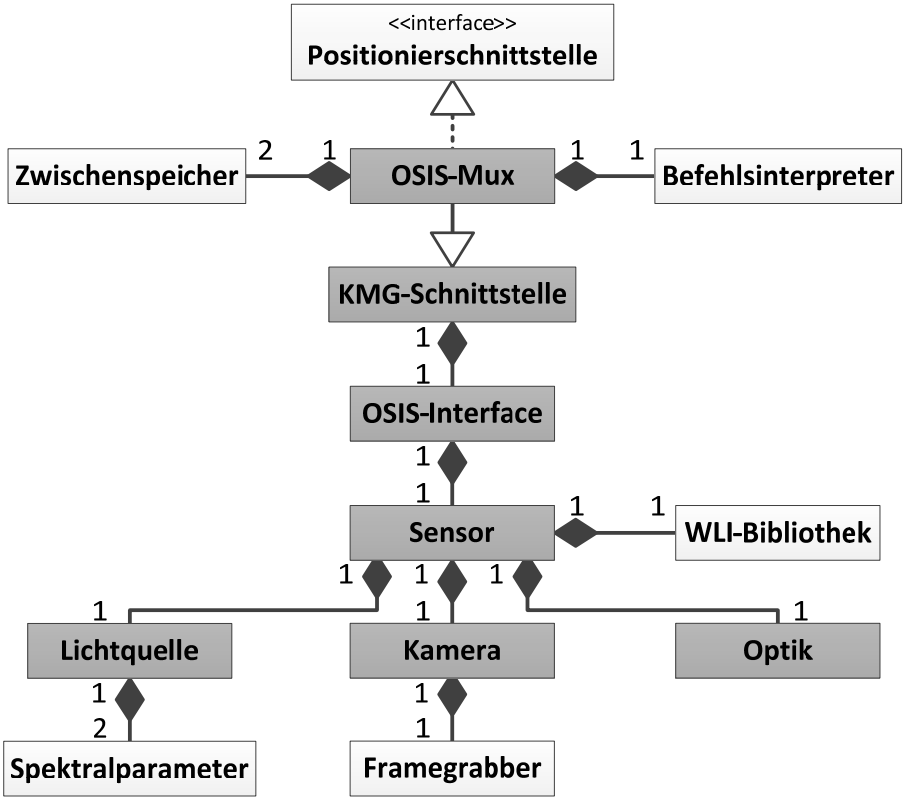


Abbildung 57: Funktionelle Bestandteile der OSIS-Implementierung des WLI-Sensors

Die Klasse der entfernt ausgeführten Instanzen, Sensor, Lichtquelle und Optik wurden um die in Abbildung 57 hell dargestellten Komponenten zur Realisierung der Datenverarbeitung und zur Speicherung der dafür notwendigen Lichtquellenparameter sowie einen Framegrabber zur Interaktion mit dem Kamerasensor erweitert.

Damit der OSIS-Multiplexer gemäß des in Kapitel 4.4.2.1 vorgestellten Konzepts die Daten optischer Sensoren mit der räumlichen Lage des Positioniersystems fusionieren kann, erbt er von der Klasse OSIS-Proxy des OSIS-Rahmenwerks den Zugang zu den Komponenten des optischen Sensors und realisiert zusätzlich die Interaktionsfunktionen der Positionierschnittstelle. Diese Realisierung bildet die Verbindung zur zentralen NPMM-Steuerung und alle darüber erfolgenden Aufrufe werden an den Befehlsinterpreter delegiert. Die Aufgabe des Befehlsinterpreters ist die Trennung der Aufrufe über die Maschinenschnittstelle in Bewegungs- und Konfigurationsbefehle, welche direkt an das subsequente Positionier-Plugin weitergeleitet werden, sowie Messbefehle, welche entsprechend des vom Sensor geforderten Messregimes umgeformt werden müssen.[6] Für die Pufferung der Bildaufnahmepositionen während einer Messung und die Speicherung der mit der WLI-Bibliothek ermittelten 3D-Daten bis zur Übertragung an die zentrale NPMM-Steuerung wurden dem OSIS-Multiplexer zwei Zwischenspeicher hinzugefügt.

[6] Beispielsweise ist eine Punktmessung durch den Befehlsinterpreter zu einer Scanbewegung entlang der optischen Achse des WLI-Sensors umzuformen. Ferner muss der Interpreter die Bildaufnahmepositionen von der NPM-Maschine abholen und den WLI-Algorithmen bereitstellen.

Die Sensorsteuerung wurde konzipiert, um auf eingebetteten System wie Smartkameras lauffähig zu sein, welche dem Nutzer nur sehr beschränkte oder keine Möglichkeiten zur Interaktion ermöglichen. Da die Sensorsteuerung im Rahmen der experimentellen Untersuchungen jedoch auf einem PC ausgeführt werden soll, wurde zusätzlich eine graphische Nutzeroberfläche erstellt (siehe Abbildung 58). Der Dialog ermöglicht eine einfache Interaktion mit der Sensorsteuerung, dient der Visualisierung von Systemzuständen und der Überwachung des Messprozesses. Die zwei großen Bildanzeigefelder stellen die letzten an die WLI-Bibliothek übertragenen Rohdaten dar und visualisieren bereits daraus ermittelte 3D-Daten. Im 3D-Daten-Fenster können einzelne Pixel mit der Maus ausgewählt werden und dessen Daten (das Interferogramm, die Antwort des Matched-Filters, die daran angepasste Zielfunktion) im darunter befindlichen Diagramm dargestellt werden. Über die Buttons können Einstellungen von wichtigen OSIS-Parametern und Parameter der Kamera angepasst oder automatische Adaptions- bzw. Kalibrierprozesse gestartet werden.

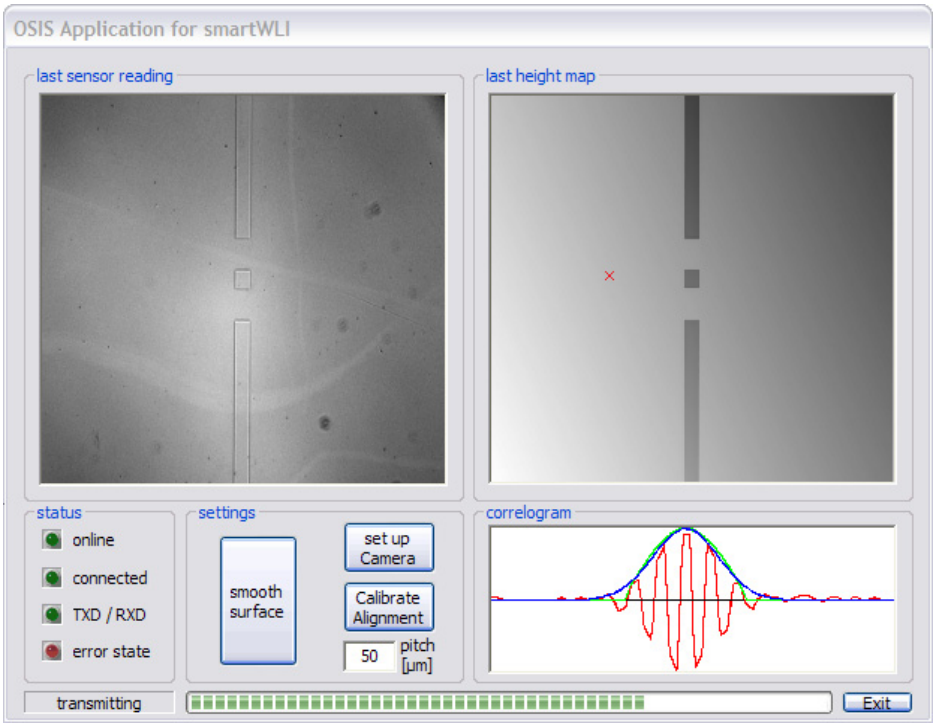


Abbildung 58: Graphische Nutzerschnittstelle der WLI-Sensorsteuerung

5.2 Der 2D-Übersichtssensor

Mit dem 2D-Übersichtssensor wird dem Anwender die Möglichkeit zur Erstellung einer ausgedehnten planaren Ansicht vom Messvolumen der NPM-Maschine bereitgestellt. Diese kann z. B. zur Identifikation interessanter Bereiche und zur Planung des optimalen Messablaufs genutzt werden.

Aufgrund der beschränkten Pixelanzahl des Sensorarrays ist es nicht möglich, den kompletten Messbereich der NPM-Maschine hoch aufgelöst im Ganzen zu erfassen. Stattdessen ist es erforderlich, Einzelaufnahmen mit hoher lateraler Auflösung, aber geringer Ausdehnung hochgenau nebeneinander zu erfassen und per Stitching zusammenzusetzen. Die geringe Positionierunsicherheit von NPM-Maschinen erlaubt das genaue Anfahren der Messpositionen, so dass die Einzelaufnahmen bei korrekter Positionierung ohne Fehler zu einer Gesamtansicht gestitcht werden können.

5.2.1 Strukturierung der Sensorsoftware

Das zentrale Element der Softwarestruktur (siehe Abbildung 59) ist, wie bei der Softwarestruktur des WLI-Sensors, die Ablaufsteuerung. Sie analysiert die Parameter eines Messaufrufs und leitet die Parameter zur Generierung der Messtrajektorie an den Positionsgenerator weiter, welcher den Sensor über die von der zentralen NPMM-Steuerung bereitgestellten Callback-Schnittstelle, die zentrale Hardwarebeschreibung und das Positionierplugin repositioniert. Der Rohdateneinzug wird beim Erreichen der vom Positionsgenerator vorgegebenen Position von der Ablaufsteuerung veranlasst. Dazu wird vom Framegrabber eine Datenaufzeichnung durch die Hardware ausgelöst und die erfassten Daten daraufhin an den Framegrabber übertragen. Anschließend werden etwaige Verzeichnungsfehler der Rohdaten korrigiert und dem Stitchingalgorithmus zugeführt. Dieser kombiniert alle auf der Messtrajektorie aufgezeichneten Einzelaufnahmen und stellt diese über den Ausgabepuffer der NPMM-Steuerung und damit dem Anwender bereit.

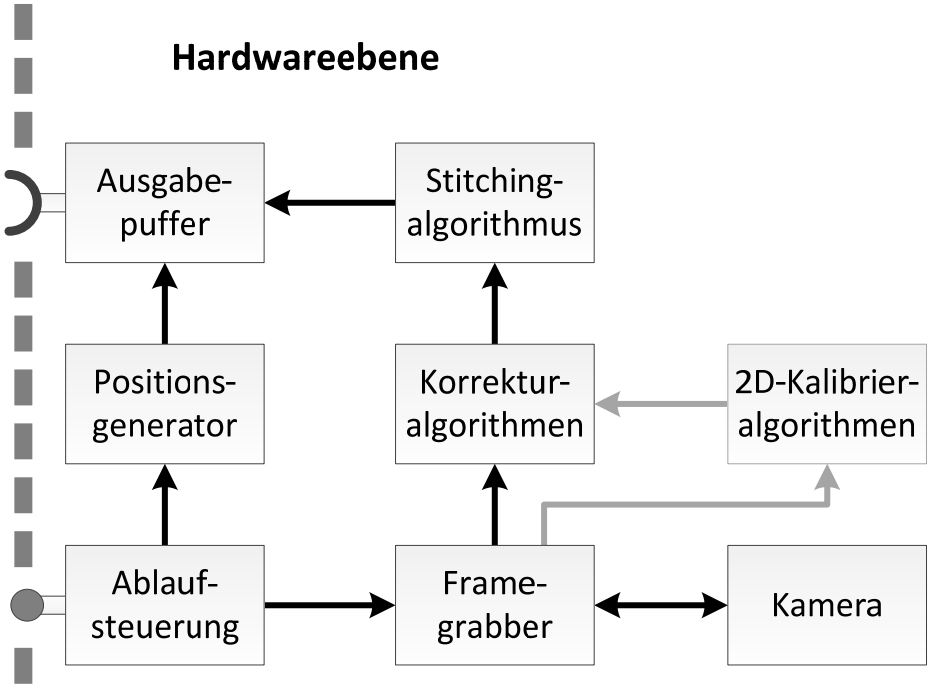


Abbildung 59: Konzept zur Strukturierung der Software des 2D-Übersichtssensors

Die für die Ermittlung der Verzeichnungsparameter erforderlichen Kalibrieralgorithmen nehmen die Stellung einer eher lose gekoppelten Komponente ein, welche nicht zwingend für den Messbetrieb erforderlich ist. Die Verzeichnungsparameter stellen im eigentlichen Sinn keine zeitlich veränderlichen Größen dar, allerdings kann es durch plastische Verformung aufgrund unsachgemäßer Handhabung, Lagerung oder durch Kollisionen während der Messung zur Veränderung einzelner Parameter kommen. Basierend auf der openCV Bildverarbeitungsbibliothek wird eine zusätzliche Kalibrierkomponente entworfen, welche alle erforderlichen Parameter bestimmt. Im Rahmen eines Kalibriervorgangs werden die Rohdaten an diese Komponente umgeleitet und die ermittelten Verzeichnungsparameter nach Abschluss an die Korrekturkomponente übergeben.

5.2.2 Aufbau der Sensorhardware

Der hardwareseitige Aufbau des optischen 2D-Überischtssensors ist in Abbildung 60 dargestellt. Er ist nahezu identisch mit dem Aufbau des WLI-Sensors. Der Unterschied besteht lediglich in der Verwendung eines konventionellen Mikroskopobjektivs anstelle des Interfernzobjektivs. Es ist es ohne weiteres möglich, den WLI-Aufbau auf die neue Aufgabe umzurüsten und diesen für die Erstellung

der 2D-Übersicht zu verwenden. Der Objektivwechsel kann entweder manuell erfolgen oder ist perspektivisch auch durch einen automatisierten Objektivrevolver realisierbar. Zudem stellt die koaxiale Auflichtbeleuchtung des Aufbaus die abschattungsfreie Ausleuchtung des Messobjekts sicher.

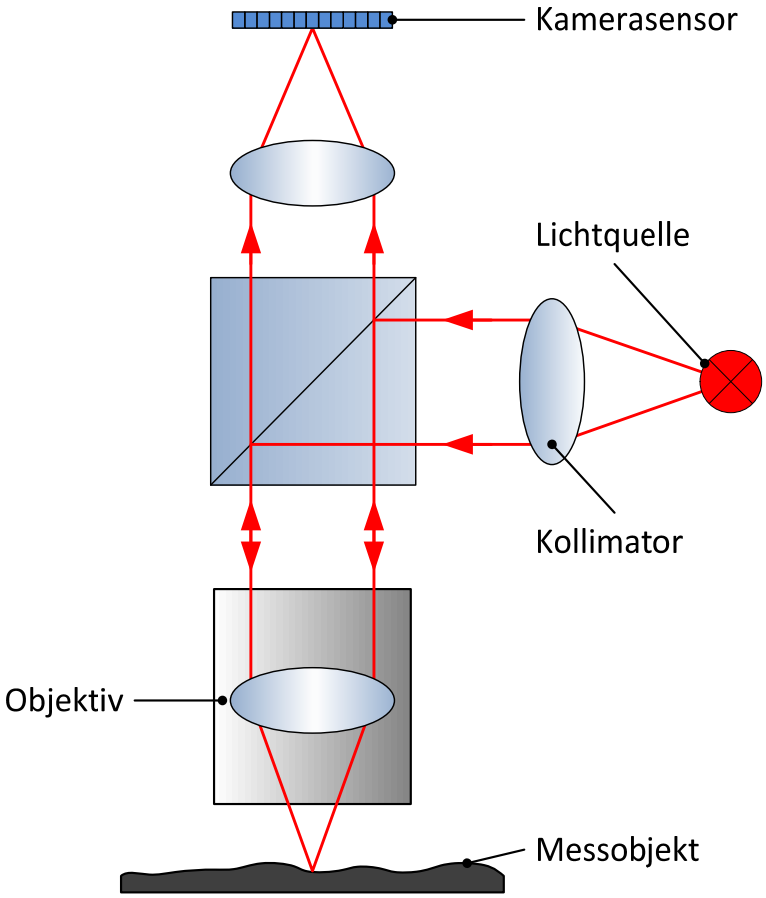


Abbildung 60: Aufbau des 2D-Übersichtssensors

5.2.3 Entwurf der Stitching-, Kalibrier- und Korrekturkomponenten

Der algorithmische Kern der Kalibrier-, Stitching- und Korrekturverfahren basiert auf den im vorigen Kapitel dargelegten Modellen und Algorithmen. Für die Implementierung der mit den Sensordaten interagierenden Komponenten wird auf die openCV Bibliothek [109] zurückgegriffen, um mit kurzer Entwicklungszeit eine performante und funktionale Lösung zu erhalten. Das Kernstück dieser Verarbeitungskette stellt der objektorientierte Framegrabber dar, welcher die Rohdaten über die Treiberschnittstelle der verwendeten Kamera einzieht und zu einem openCV konformen Datenobjekt konvertieret.

5.2.4 Integration der Komponenten des 2D-Übersichtssensors in den OSIS-Kontext

Der Aufbau der Ansteuerungssoftware des 2D-Übersichtssensors folgt einem ähnlichen Konzept wie der Entwurf des WLI-Sensors. Jedoch wird wegen der fehlenden Unterstützung für Flächendaten nicht auf das OSIS-Rahmenwerk zurückgegriffen. Als Interaktionsschnittstelle wird stattdessen die proprietäre Schnittstelle für optische Parallelsensoren verwendet. Auch bei diesem Entwurf werden die Komponenten der Sensorsteuerung in zwei dezentrale Datenverarbeitungskomponenten aufgegliedert. Die für den Aufbau der Sensorsteuerung erforderlichen Komponenten sind in Abbildung 61 veranschaulicht.

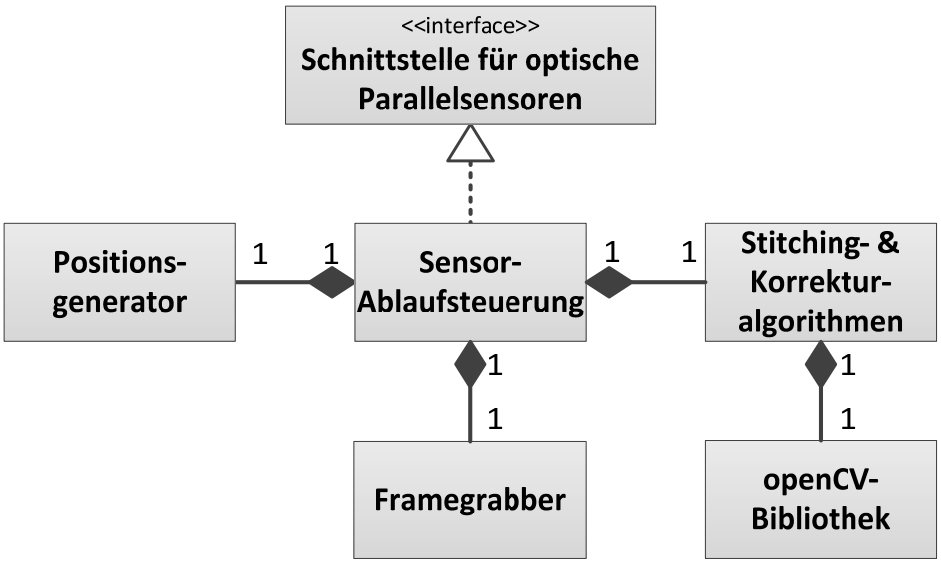


Abbildung 61: Funktionelle Bestandteile der OSIS-Implementierung des 2D-Übsichtssensors

Die zentrale Instanz der Steuerungssoftware des 2D-Übersichtssensors ist die Sensorablaufsteuerung, welcher alle anderen Instanzen untergeordnet sind. Im Rahmen der Datenerfassung koordiniert die Ablaufsteuerung alle subsequenten Instanzen.

Durch den Positionsgenerator werden Daten für die Steuerung des Positioniersystems erzeugt, welche über die Schnittstelle an das Positioniersystem weitergegeben werden. Die Aufzeichnung von Rohdaten erfolgt durch den Framegrabber, welcher die Rohdaten für die geometrische Korrektur unter Anwendung der Korrekturalgorithmen, basierend auf der openCV-Bibliothek, an die Ablaufsteuerung weiterreicht. Alle erfassten Rohdaten werden von der Ablaufsteuerung gepuffert und abschließend durch die Stitchingalgorithmen zu einer umfassenden Gesamtdarstellung zusammengefügt.

Die Steuerungssoftware des 2D-Übersichtssensors ist wie die Steuerung des WLI-Sensors für die Datenverarbeitung ohne Nutzerinteraktion konzipiert. Im Gegensatz zu diesem wurde für den 2D-Übersichtssensors jedoch keine optionale Benutzerschnittstelle für die Überwachung und Analyse der Steuerungssoftware umgesetzt.

5.3 Der optische 3D-Sensor

Um die Messstrategie an räumlich ausgedehnten Objekten im Vorfeld einer Messung planen und optimieren zu können, sollen mit einem dafür geeigneten Sensor hinreichend aufgelöste dreidimensionale Primärdaten erzeugt und dem Anwender für die Planung bereitgestellt werden. Um die Messdauer für die Erzeugung der 3D-Punktewolke gering zu halten, soll der 3D-Übersichtssensor auf einem triangulationsbasierten Stereovision-Verfahren basieren.

5.3.1 Strukturierung der Sensorsoftware

Die Softwarestruktur des optischen 3D-Sensors (siehe Abbildung 62) orientiert sich am gleichen Konzept wie die Entwürfe der übrigen Sensoren. Die internen Datenerfassungs- und -verarbeitungsabläufe werden ebenfalls von einer internen Ablaufsteuerung organisiert, welche die Steuerung und die Datenerfassung über den Framegrabber veranlasst. Die gewonnenen Rohdaten werden auch beim 3D-Sensor algorithmisch korrigiert, bevor sie an die eigentliche 3D-Datenverarbeitungseinheit weitergereicht werden, welche daraus eine 3D-Punktewolke extrahiert und

diese für die Ausgabe an die zentrale NPMM-Steuerung bzw. den Anwender im Ausgabepuffer sendet.

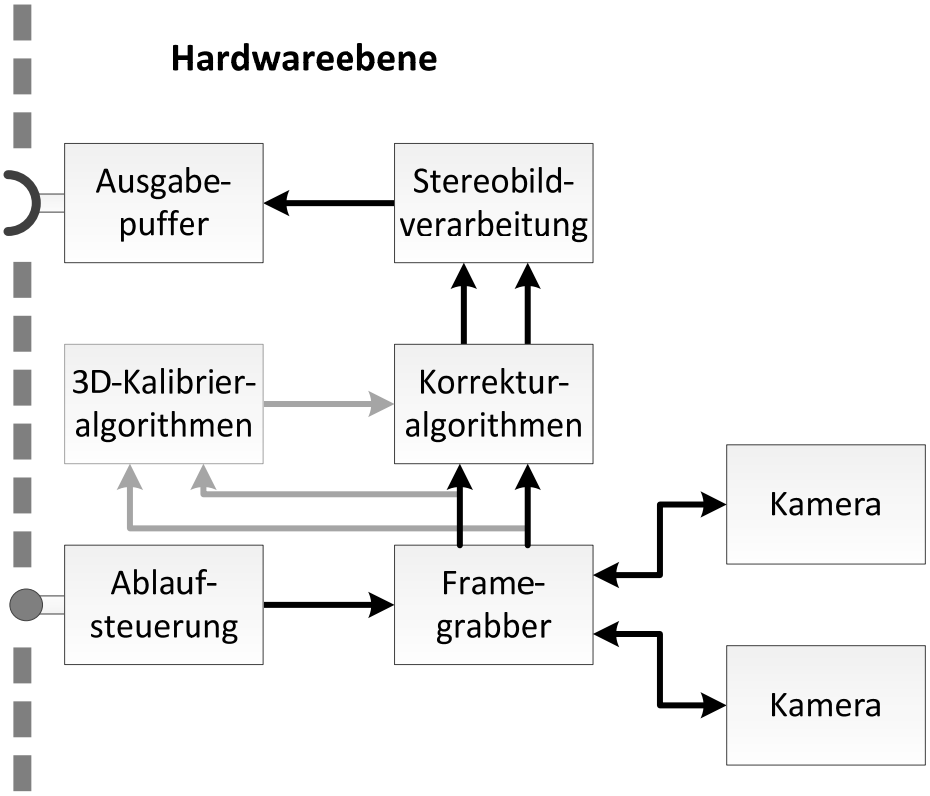


Abbildung 62: Konzept zur Strukturierung der Software des optischen 3D-Sensors

Die hardwareabhängigen Parameter für die Korrektur der Rohdaten behalten, insofern keine Dejustierung des Sensoraufbaus z. B. durch starke Erschütterungen auftritt, über einen langen Zeitraum ihre Gültigkeit. Daher ist es hinreichend, die Parameter in einem gesonderten Offlinekalibrierverfahren zu ermitteln und für die spätere Verwendung persistent zu speichern.

5.3.2 Aufbau der Sensorhardware

Der hardwareseitige Sensoraufbau ist vergleichsweise einfach gehalten. Er besteht aus zwei mit je einem Objektiv versehenen Industriekameras, welche auf einer gemeinsamen Grundplatte montiert sind. Die Ausrichtung der optischen Achsen beider Kamerasysteme ist nahezu parallel und wird durch eine Führung in der Grundplatte vorgegeben. Die Position der Kameras ist entlang der Führung variabel, wodurch der Basisabstand und damit die Tiefenauflösung des Sensorsystems variiert werden kann. Die schematische Darstellung in Abbildung 63 zeigt den kompletten Aufbau, wobei die optischen Achsen nicht zwangsläufig senkrecht zur Grundebene des Positioniersystems auszurichten sind. Prinzipiell kann die Grundplatte beliebig am metrologischen Rahmen des Positioniersystems befestigt werden. Eine orthogonale Ausrichtung der optischen Achse zur Positionier-Grundebene verringert jedoch die Wahrscheinlichkeit von Abschattungen, da bei dieser Konstellation räumlich ausgedehnte Messobjekte etwaige Teilbereiche des Messvolumens nicht verdecken.

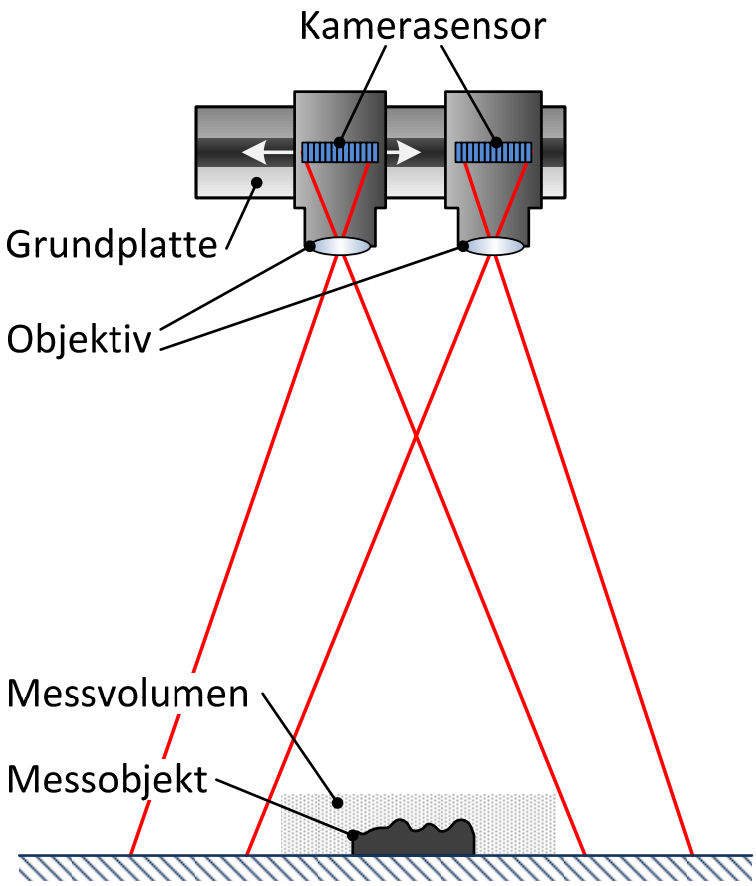


Abbildung 63: Aufbau und Anordnung des 3D Übersichtssensors

5.3.3 Entwurf der Stereobildverarbeitungs-, Kalibrier- und 3D Korrekturkomponenten

Da es sich bei allen Datenverarbeitungsaufgaben um Aufgabenstellungen aus dem Bereich der klassischen Bildverarbeitung handelt, kann für die Realisierung der Kalibrier-, Korrektur- und Stereobildverarbeitung auf die existierenden Lösungen der openCV zurückgegriffen werden. Die Algorithmen der openCV setzen die im vorigen Kapitel eingeführten modellbasierten Verfahren um, sind mit besonderer Rücksicht auf einen hohen Datendurchsatz implementiert und erlauben durch den hohen Softwarereifegrad eine schnelle, effiziente und fehlersichere Entwicklung der zu implementierenden Softwarekomponenten des 3D-Sensors.

5.3.4 Integration der Komponenten des 3D-Sensors in den OSIS-Kontext

Der Aufbau der Ansteuerungssoftware des 3D-Sensors folgt dem gleichen Konzept wie der Entwurf des 2D-Übsichtssensors. Als Interaktionsschnittstelle wird ebenfalls die proprietäre Schnittstelle für optische Parallelsensoren verwendet und die Komponenten der Sensorsteuerung auch bei diesem Entwurf in zwei dezentrale Datenverarbeitungskomponenten aufgegliedert. Die für den Aufbau der Sensorsteuerung erforderlichen Komponenten sind in Abbildung 64 veranschaulicht.

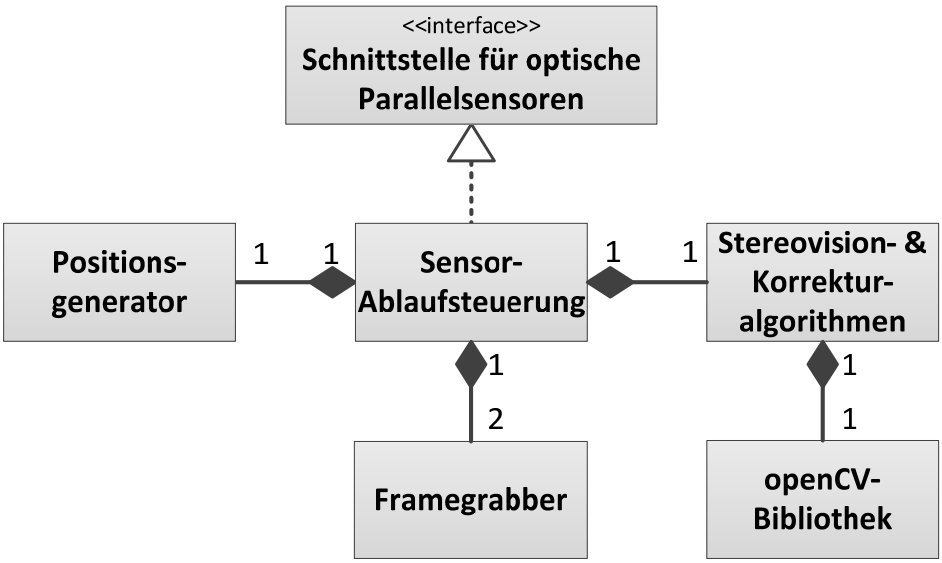


Abbildung 64: Funktionelle Bestandteile der OSIS-Implementierung des 3D-Sensors

Die zentrale Instanz der Steuerungssoftware des 3D-Sensors ist die Sensorablaufsteuerung, welcher alle anderen Instanzen untergeordnet sind. Im Rahmen der Datenerfassung werden diese Instanzen von der Ablaufsteuerung koordiniert. Durch den Positionsgenerator werden Daten für die Steuerung des Positioniersystems erzeugt und Rohdaten über die beiden Framegrabber aufgezeichnet. Die Rohdaten werden unter Anwendung der Korrekturalgorithmen durch die Ablaufsteuerung entzerrt und über die auf der openCV-Bibliothek basierenden Stereovisionalgorithmen zu einer 3D-Punktewolke weiterverarbeitet.

Die Steuerungssoftware des 3D-Übersichtssensors ist wie die Steuerung des 2D-Übersichtssensors für die Datenverarbeitung ohne Nutzerinteraktion konzipiert und es wurde keine optionale Benutzerschnittstelle für die Überwachung und Analyse der Steuerungssoftware umgesetzt.

Damit ist der Entwurfs- und Implementierungsprozess für die modulare schnittstellenbasierte NPMM-Steuerung abgeschlossen. Auf Grundlage der vorgestellten Konzepte wurde ein Demonstrator der NPMM-Steuerung entwickelt, welcher über verschiedene Erweiterungen das Positioniersystem NMM-1 sowie optische und taktile Sensoren zu einem Gesamtsystem integriert. Der globale Funktionsnachweis des Gesamtsystems sowie die Analyse der Leistungsfähigkeit einzelner Komponenten wird im folgenden Kapitel eingehend dargelegt.

Kapitel 6 Experimentelle Untersuchungen

Im Anschluss an den Entwurf der Software für die optischen Sensorsysteme sollen das Laufzeitverhalten und die algorithmisch korrekte Funktion des Demonstrators der NPMM-Steuerung sowie der zugehörigen Steuerungsplugins - darunter das Positionierplugin für die NMM-1, die Erweiterungen für optische Sensoren sowie die Rohdatenverarbeitungsbibliotheken für WLI-Daten - analysiert werden. Mit der zunächst durchzuführenden Untersuchung des Laufzeitverhaltens sollen etwaige Verarbeitungsengpässe, welche das Erreichen der in Kapitel 2 gesetzten Zielparameter für die Messfrequenz beinträchtigen, aufgedeckt werden. Soweit möglich werden diese Engpässe beseitigt oder mögliche Ansätze für ihre Behebung erörtert.

Im zweiten Abschnitt dieses Kapitels wird die Demonstratorimplementierung durch experimentelle Messungen mit optischen und taktilen Sensoren an verschiedenen Objekten funktionell validiert. Die Zielstellung dieser Untersuchungen ist der abschließende Funktionsnachweis der dynamisch skalierbaren, heterogen teilparallelisierten und auf mehrere Datenverarbeitungssysteme verteilten Software. Vom direkten Nachweis metrologischer Eigenschaften durch entsprechende Messungen an geeigneten Normalen wurde abgesehen, da diese vornehmlich von den Eigenschaften der für die Datenerfassung eingesetzten Sensorhardware und nur im vernachlässigbarem Maße von der Datenverarbeitungslogik der NPMM-Steuerung beeinflusst wird.

6.1 Abschätzung des numerischen Fehlers

Die numerische Verarbeitung der metrologischen Daten beschränkt sich auf die Transformation der Rohdaten aus dem intrinsischen Koordinatensystem des Positionierers in ein vom Anwender frei wählbares Weltkoordinatensystem. Bei der Berechnung der Zielkoordinaten treten aufgrund der endlichen Rechengenauigkeit digitaler Datenverarbeitungssysteme zwangsläufig Rundungsfehler auf, deren Auswirkung auf den Gesamtmessfehler im Folgenden analysiert werden soll.

Der Einfluss auf die metrologische Reproduzierbarkeit stellt sich wie folgt dar:

$$f(\vec{x}) = \vec{y} = R \cdot \vec{x} + \vec{t} \text{ in } \boldsymbol{R}^3 \tag{6.1}$$

das Ergebnis $\vec{y}$ wird aus drei Produkten und ihrer anschließenden Summation inklusive der Verschiebungskomponente $\vec{t}$ berechnet. Die einzelnen Komponenten i des Ausgangsvektors berechnen sich somit wie folgt:

$$y_i = t_i + \sum_{j=1}^{3} r_{i,j} \cdot x_j \tag{6.2}$$

Bereits in Abschnitt 4.3.2 wurde die numerische Genauigkeit der zentralen Datenverarbeitungslogik erörtert. Da digitale Datenverarbeitungssysteme reelle Zahlen nicht beliebig genau darstellen können, ist jeder Wert in (6.3) mit einem relativen Fehler in der Höhe der Maschinengenauigkeit ε_{rel} behaftet.

$$y_i = \sum_{j=1}^{3} \left(r_{i,j}(1 + \varepsilon_{rel}) \cdot x_j(1 + \varepsilon_{rel}) \right) + t_i(1 + \varepsilon_{rel}) \tag{6.3}$$

Die nach Auflösung der binomischen Formeln entstehenden Terme mit quadratischem Fehleranteil sind vernachlässigbar klein und werden nicht weiter betrachtet, wodurch sich (6.3) wie folgt vereinfachen lässt:

$$y_i = \sum_{j=1}^{3} \left(r_{i,j} x_j \cdot (1 + 2\varepsilon_{rel}) \right) + t_i(1 + \varepsilon_{rel}) \tag{6.4}$$

Durch die Multiplikation wird der relative Fehler somit verdoppelt. Die weitere analytische Auflösung von (6.4) ermöglicht eine Darstellung zur Ermittlung des durch die Berechnung resultierenden relativen Fehlers Δy_i wie folgt:

$$y_i = \sum_{j=1}^{3} \left(r_{i,j} x_j \right) + t_i + \sum_{j=1}^{3} \left(2 r_{i,j} x_j \varepsilon_{rel} \right) + t_i \varepsilon_{rel} \tag{6.5}$$

$$\Delta y_i = \frac{\sum_{j=1}^{3} (2 r_{i,j} x_j \varepsilon_{rel}) + t_i \varepsilon_{rel}}{\sum_{j=1}^{3} (r_{i,j} x_j) + t_i} \text{ mit } \sum_{j=1}^{3} |r_{i,j}| = 1 \tag{6.6}$$

Der letztendlich resultierende Fehler bestimmt sich also maßgeblich aus den konkreten Größen für die affine Transformation und den Eingangsvektor. Aus dem Gauß'schen Fehlerfortpflanzungsgesetz kann jedoch geschlussfolgert werden, dass der mittlere Fehler im Regelfall unterhalb der durch die Produktbildung hervorgerufenen doppelten Maschinengenauigkeit von $\varepsilon_{rel} \approx 2{,}2 \cdot 10^{-16}$ m liegen wird.

Im theoretischen Sinn ist lediglich der Fall der numerischen Auslöschung kritisch. Bei der betragsmäßigen Addition im Bereich der Darstellbarkeit vergleichbarer Summanden mit entgegengesetztem Vorzeichen kann der Fehler sprunghaft ansteigen, wenn sich die Summanden gegenseitig im Extremfall nahezu vollständig aufheben. Dieser Fall kann aber aufgrund des Quantisierungsrauschens einer NPM-Maschine von $\varepsilon_{pos} = 5 \cdot 10^{-12}$ m im Vergleich zum numerischen Darstellungsfehler ($\varepsilon_{rel} = 1{,}1 \cdot 10^{-16}$m) nahezu vollständig ausgeschlossen werden.

Wie dargelegt wurde, spielt der numerische Fehler bei der Verarbeitung metrologischer Daten eine vernachlässigbar geringe Rolle. Die dominanten Größen für den Gesamtfehler bei einer Koordinatenberechnung sind sowohl die Reproduzierbarkeit des Sensorsystems, welche sich üblicherweise im mittleren einstelligen Nanometerbereich bewegt, als auch das Quantisierungsrauschen des Positioniersystems.

6.2 Analyse des Laufzeitverhaltens der NPMM-Steuerung

6.2.1 Modellierung des erwarteten Laufzeitverhaltens

Wegen des hohen Datenaufkommens bei nanometrologischen Applikationen liegt der Fokus der ersten Untersuchungen am Demonstrator der NPMM-Steuerung besonders auf der Laufzeiteffizienz der Implementierungen. Die Konvertierung und Übertragung der Messdaten durch die NPMM-Steuerung muss aufgrund der hohen Messfrequenz der NPM-Maschine mit dem erzeugten Datenaufkommen mithalten können. Dieser Nachweis wird im Folgenden durch entsprechende Laufzeitmessungen erbracht. Dazu werden am Demonstrator die Kenngrößen Latenz und Durchsatz für die Verarbeitung der metrologischen Daten ermittelt. Beide Faktoren wirken sich in Kombination als eine Superposition von Verzögerungszeiten auf die zu verarbeitenden Messdaten aus (siehe Abbildung 65).

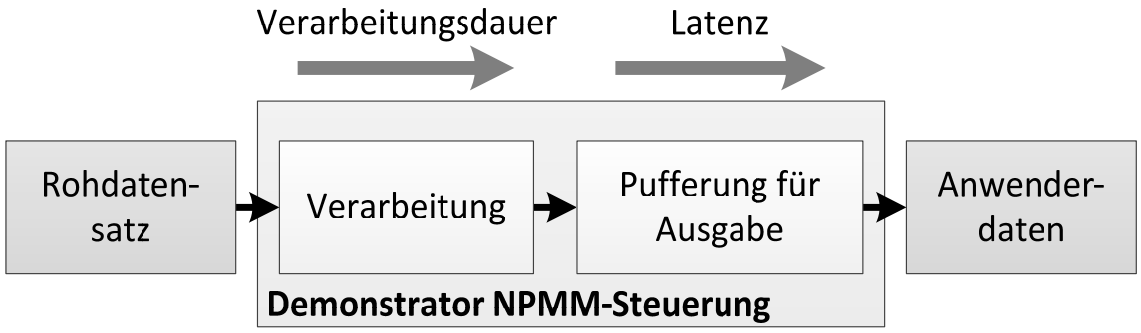


Abbildung 65: Modell der Messdatenverzögerung am NPMM-Demonstrator

Hierbei beschreibt der Datendurchsatz bzw. sein Kehrwert die Verarbeitungsdauer, wie viele Messdatensätze pro Zeiteinheit verarbeitet und transportiert werden können und steht damit stellvertretend für die Geschwindigkeit des Systems. Der Durchsatz ist sowohl von Umfang und Effizienz der Datenverarbeitung als auch von der CPU-Geschwindigkeit des NPMM-Steuerungssystems abhängig und stellt einen limitierenden Faktor im Hinblick auf die erreichbare Messgeschwindigkeit dar. Erreicht der Durchsatz nur geringe Größen, kann die maximale Messgeschwindigkeit der metrologischen Hardware nicht vollständig ausgereizt werden, da es anderenfalls aufgrund eines Rückstaus der Rohdaten unweigerlich zu Pufferüberläufen und damit zu einem Messdatenverlust kommt. Die Latenz beschreibt dagegen die Dauer vom Auftreten der Daten am Messsysteminterface bis zur Weitergabe über die Anwenderschnittstelle und wird durch die Verweildauer in Puffern und Warteschlangen bestimmt. Sie kann als Totzeit oder Trägheit des Systems verstanden werden und ist in Bezug auf einen potentiellen Datenverlust allerdings ohne maßgebliche Bedeutung.

6.2.2 Spezifikation der Testumgebung

Um die Leistungsfähigkeit der NPMM-Steuerung realistisch bewerten zu können, wurde die Untersuchung auf der beabsichtigten Zielhardware (siehe Tabelle 17) und einem üblichen Messszenario durchgeführt.

Komponente	Name & Parameter
Prozessor	Intel Core 2 Duo E8600 (2x 3,33 GHz)
Hauptspeicher	2x Kingston 2GB DDR2 (Takt 400Mhz, Latenz 5-5-5-18)
Motherboard	ASUS P5QL-E (Chipsatz: Intel P43 + ICH10R)

Tabelle 17: Datenverarbeitungshardware des Demonstrators

Das Messdatenverarbeitungsszenario umfasst die Verrechnung der Messdaten vom Positioniersystem und der Sensorhardware sowie die Ausgabe der berechneten Koordinaten über die Anwenderschnittstelle. Typischerweise umfassen die Rohdaten die drei räumlichen Positionsdaten der Achseninterferometer sowie die 3D-Auslenkung des Tastsystems. NPM-Maschinen können darüber hinaus noch weitere Rohdaten bereitstellen. Da die NPMM-Steuerung jedoch mit Zielsetzung auf die generische I++DME Schnittstelle entworfen wurde, müssen diese Daten nicht berücksichtigt werden. Das Datenaufkommen zwischen Hardwareebene und NPMM-Steuerung beläuft sich damit auf drei Fließkommawerte mit doppelter und drei weitere Werte mit einfacher Genauigkeit. In der Summe umfasst ein Messdatensatz somit 36 Byte. Die Erzeugung der Rohdaten wird nicht mit realer Hardware durchgeführt. Vielmehr werden geeignete Simualtionsplugins verwendet, welche über die Schnittstellen der Hardwareebene eingebunden werden.

6.2.3 Latenz der NPMM-Steuerung

Die Erfassung der zu untersuchenden Messdatenverzögerung erfolgt mithilfe zweier Systemfunktionen der *WinAPI.* Über die Funktionen *QueryPerformanceCounter* und *QueryPerformanceFrequency* kann für die Zeitmessungen im Mikrosekundenbereich auf einen hochpräzisen Systemtaktzähler zugegriffen werden. Diese API-Funktionen wurden in einer zusätzlichen Profilerklasse für den Einsatz in der NPMM-Steuerung zusammengefasst. Die Verarbeitungslogik der Profilerklasse wurde mit besonderer Rücksicht darauf entworfen, die Abläufe in der NPMM-Steuerung zeitlich nicht zu beeinflussen und die Zeitmessung mit minimalen Änderungen in die NPMM-Steuerung zu integrieren und die gewonnenen Daten nach Abschluss der Messung automatisch als auswertbare Datei bereitzustellen.

Für die Erzeugung der Rohdaten von Positionier- und Sensorsystemen wurde in einem ersten Versuch ein für die Entwicklung der NPMM-Steuerung entworfener NMM-1 Simulator verwendet. Dieser wird anstelle der Steuerungsplugins für reale Hardware von der NPMM-Steuerung geladen und ist in der Lage, sämtliche Positionier und Antastsequenzen für die NPMM-Steuerung transparent nachzubilden. Um dynamisch auf die von der NPMM-Steuerung angeforderten Sequenzen reagieren zu können, verfügt der Simulator über einen umfangreichen Algorithmus, welcher die Simulationsdaten während der Laufzeit berechnet (vgl. Abschnitt 4.4.1.2). Der mit diesem Ansatz gemessene zeitliche Verlauf der Messdatenlatenzen ist in Abbildung 66 dargestellt.

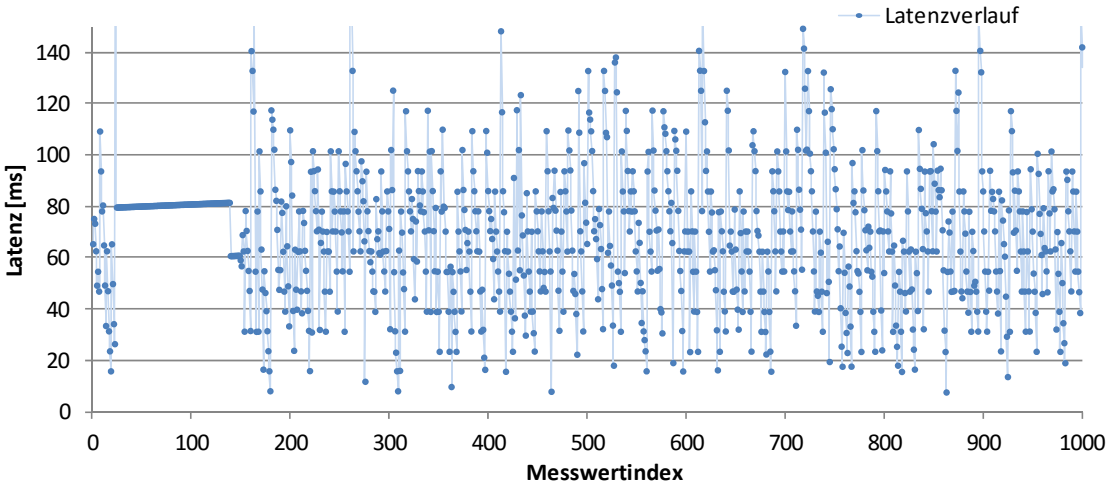


Abbildung 66: Latenz je Datenpaket mit Softwaresimulator

Die gemessene mittlere Latenz beträgt

$$\bar{L} = 69{,}37\ ms.$$

Jedoch spiegelt die gemessene Latenz nicht die tatsächliche Leitungsfähigkeit der NPMM-Steuerung wider. Die offensichtliche Schichtung der Latenzen in Quanten von circa 8 ms deutet auf ein Datentransportproblem hin. Die Ursache dafür ist im Scheduling des Betriebssystems und der komplexen Messdatenerzeugung im Simulator zu suchen. Die nebenläufigen Leichtgewichtprozesse für die Erzeugung der Rohdaten verdrängen die Messdatenverarbeitung der NPMM-Steuerung durch ihr hohes Datenverarbeitungsaufkommen sehr häufig von der CPU. Der Datenübertragungsalgorithmus der NPMM-Steuerung erhält damit nicht ausreichend CPU-Zeit, um die Rohdaten in einem kontinuierlichen Vorgang abarbeiten zu können, wodurch die gemessene mittlere Latenz von der Messdatenerzeugung negativ beeinflusst wird.

Als Konsequenz dieser Erkenntnis wurde der Datenverarbeitungsaufwand für die Erzeugung der Rohdaten auf ein absolutes Minimum beschränkt. Anstatt die simulierten Rohdaten dynamisch zu erzeugen, wurden im zweiten Versuch statische vorabdefinierte Rohdaten an die NPMM-Steuerung übergeben, wodurch die Rechenzeit für die Messwerterzeugung vernachlässigt werden kann. Die Ergebnisse der Latenzmessung mit der adaptierten Datenerzeugung sind in Abbildung 67 dargestellt.

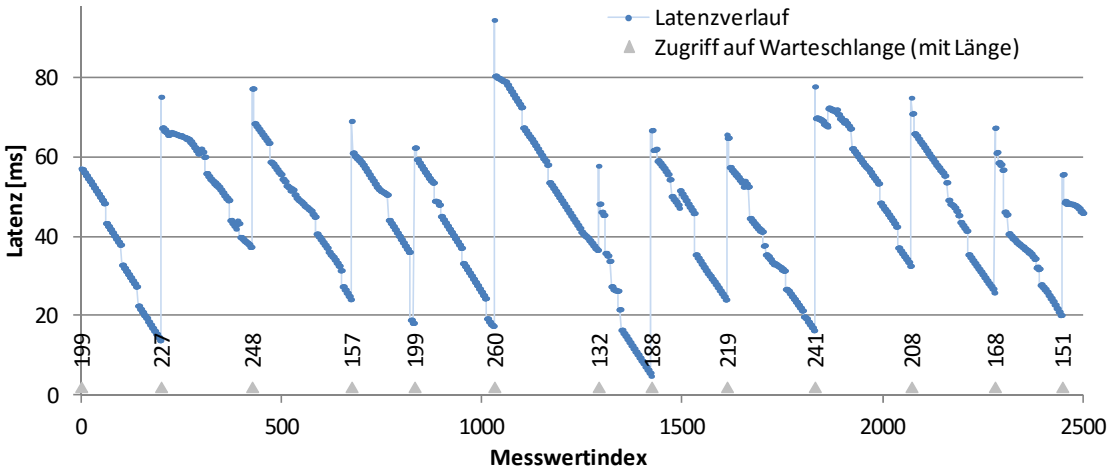


Abbildung 67: Latenz je Datenpaket bei minimaler CPU-Last für die Rohdatensynthese

Die Schichtung der Latenzen in Zeitquanten ist nicht mehr zu beobachten, wodurch die maßgebliche Behinderung der Datenverarbeitung durch die komplexe Rohdatengenerierung zur Laufzeit bestätigt wurde. Die im Rahmen der zweiten Messung ermittelte mittlere Latenz von

$$\bar{L} = 43{,}16\,ms$$

ist auch dahingehend plausibel, da der zeitlich aufgelöste Latenzverlauf einzelner Messdaten aus der Verarbeitungsarchitektur der NPMM-Steuerung erklärt werden kann.

Ein Empfangsthread nimmt die Messdaten kontinuierlich von der NPM-Maschine entgegen, konvertiert sie und legt sie in einer Warteschlange ab. Ein anderer Thread, dessen Aufgabe unter anderem die Datenübertragung zum Client ist, holt die Daten periodisch aus der Warteschlange ab und sendet sie an den Nutzer. Dadurch entsteht das in Abbildung 67 zu erkennende Sägezahnmuster. Neben dem Latenzverlauf sind die Zeitpunkte der Warteschlangenabarbeitungen mit der Anzahl der wartenden Pakete als graue Pfeilspitzen dargestellt. Die Datenpakete werden bei Eintritt in die Abarbeitung sehr schnell nacheinander versendet. Da sie jedoch kontinuierlich an die Warteschlange angefügt werden und der Versendethread mit einer längeren Periode als der Empfangsthread die Warteschlange bedient, liegen die Daten eine Zeitspanne von mehreren 10 ms vor ihrer Versendung in der Warteschlange.

Dieses Verhalten ist unproblematisch, da der Versendethread die Warteschlange stets vollständig abarbeiten kann. Die Zeitdauer für die Pufferung der Messdaten liegt bei voller Messfrequenz der NMM-1 deutlich unter 100 ms und ist damit während einer Messung vernachlässigbar klein. Sollte das System bei höherem Rohdatenaufkommen eine deutliche Verzögerung aufweisen oder die Warteschlange nicht mehr vollständig abgebaut werden, kann diesem Verhalten durch eine höhere Periodisierung des Versendethread entgegengewirkt werden.

6.2.4 Messdatendurchsatz der NPMM-Steuerung

Im Folgenden wird die Arbeitsgeschwindigkeit der NPMM-Steuerung untersucht. Da NPM-Maschinen Messfrequenzen von 6,25 kHz bzw. 83,33 kHz im Fall der NPMM-200 aufweisen, ist sicher zu stellen, dass der Server mit dem Datenaufkommen Schritt halten kann und es nicht zu einem Messdatenverlust aufgrund des beschränkten Zwischenspeichers der NPM-Maschine kommt. Für die Analyse wurde der Server einem Stresstest unterzogen, um die durchschnittliche Datenverarbeitungsfrequenz zu bestimmen.

Zu diesem Zweck wurde ein Testplugin für die Maschinenschnittstelle implementiert, welches eine NPM-Maschine mit unendlich großer Messfrequenz simuliert. Die interne Datenverarbeitung dieses Testplugins wurde auf ein absolutes Minimum beschränkt, um den Einfluss auf die Durchsatzmessung aufgrund von CPU-Ressourcenverbrauch so gering wie möglich zu halten. Da auch die Datenübertragung über das TCP/IP Netzwerk die Datenübertragung zwischen Server und Client potentiell behindern kann, muss die Clientsoftware auf demselben PC ausgeführt werden wie der Server. Die Kommunikation erfolgt in diesem Falle über das TCP/IP Protokoll, jedoch nicht über ein reales Netzwerk. Vielmehr werden die Daten vom *Loopback Interface* der Netzwerkkarte an sich selbst (zurück-)gesendet.

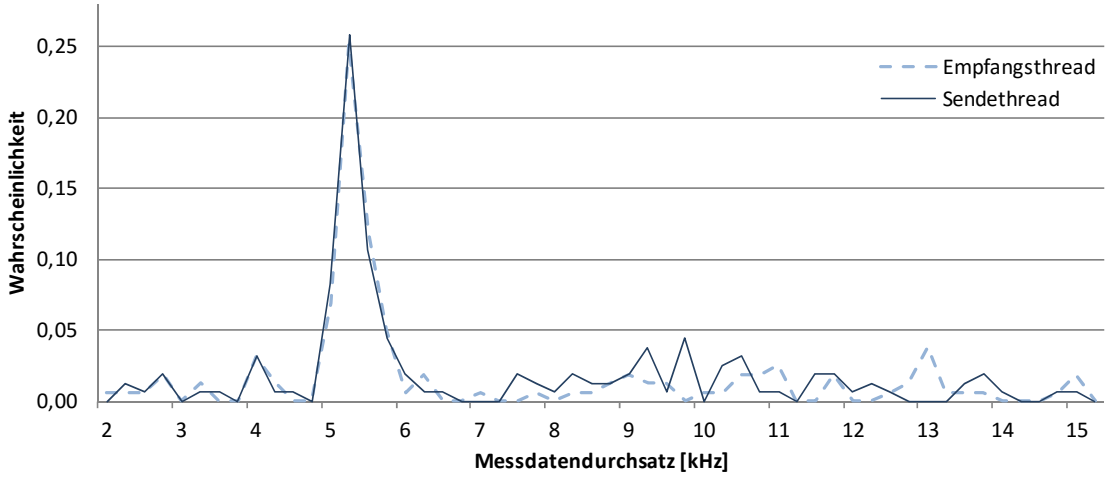


Abbildung 68: Histogramm des Datendurchsatzes

Die Ermittlung der Verarbeitungsgeschwindigkeit erfolgt gesondert für den Empfangs- und Sendethread. Gemessen wurde die Zeitspanne für das Empfangen bzw. Versenden von 160000 Messpunkten. Ein Messpunkt besteht aus seiner Raumkoordinate und einem Vektor für die Antastrichtung. Da die Kommunikation über die Maschinenschnittstelle nur durch Funktionsaufrufe realisierbar ist, wurde die Messdatenabfrage des Servers vom Testplugin mit einem Paket von 1000 Punkten beantwortet.

Das Ergebnis ist in Abbildung 68 dargestellt. Die Hälfte der Daten (ca. 52 %) wird von beiden Threads mit einer Frequenz von (5,17 ± 0,04) kHz verarbeitet. Die verbleibenden 48 % der Daten liegen näherungsweise gleichverteilt in einem Bereich zwischen 2 kHz und 16,7 kHz. Der globale Datendurchsatz beträgt im Mittel 6,92 kHz am Eingabethread (NPM-Maschine zum Server) und 7,04 kHz am Ausgabethread (Server zum Client). Damit ist die Datenverarbeitung des Servers auf dem Zielsystem hinreichend schnell, um ohne zusätzliche Puffermechanismen dauerhaft mit einer NMM-1 arbeiten zu können.

Für die direkte Nutzung einer NPMM-200 ist die Verarbeitungsgeschwindigkeit jedoch zu gering. Es bedarf eines vorgeschalteten Datenpuffers mit variabler Größe, um Daten, welche nicht sofort verarbeitet werden können, zwischenzuspeichern. Dieser Datenpuffer wird implizit durch das Maschinenplugin der NPMM-200 bereitgestellt.

6.3 Messdatendurchsatz der WLI-Algorithmen

Um die Leistungsfähigkeit der Weißlichtalgorithmen möglichst realistisch abzubilden, wurde ihre Untersuchung ebenso wie die Validierung der NPMM-Steuerung auf dem beabsichtigten Zielsystem durchgeführt. Die für den Datendurchsatz des Systems relevante Komponenten sind in Tabelle 18 aufgeführt.

Komponente	Name & Parameter
Prozessor	Intel Core 2 Quad Q9650 (4x 3,00 GHz)
Hauptspeicher	4x Kingston 2GB DDR2 (Takt 400Mhz Latenz 5-5-5-18)
Motherboard	ASUS P5QL-E (Chipsatz: Intel P43 + ICH10R)

Tabelle 18: Konfiguration des Bildverarbeitungs-PC

Die Untersuchung der WLI-Bibliothek erfolgt mit der in Abschnitt 6.2.3 beschriebene Profilerklasse. Die damit durchgeführte Zeitmessung wurde unmittelbar vor Eintritt in die WLI-Datenverarbeitungsroutine gestartet und unmittelbar nach Abschluss der Datenverarbeitung gestoppt. Als Testdatensatz diente ein a-priori aufgezeichneter Weißlichtinterferenzbildstapel mit 110 äquidistanten Aufnahmen zu jeweils 1024 mal 768 Bildpunkten mit je 8 Bit zur Quantifizierung des Interferenzkontrasts, welche über einen z-Verstellbereich von 7,7 µm aufgezeichnet wurden. Für die Bestimmung der 2,5D-Raumkoordinaten durch den Hauptverarbeitungsalgorithmus kann über einen Initialisierungsparameter ein Regressionsmodell gewählt werden. Es wurden Messreihen zu jeweils 50 Wiederholungen durchgeführt und die in Tabelle 19 präsentierten Ergebnisse auf dem Zielsystem ermittelt.

Regressionsmodell	Verarbeitungsdauer [s]	Güte
Gaußverteilung	12,20 ± 0,16	sehr hoch
Parabel	2,92 ± 0,01	hoch
Maximumsuche	2,19 ± 0,08	gering

Tabelle 19: erzielte Verarbeitungsdauer für WLI-Messdaten

Die drei Hauptverarbeitungsmethoden mit ihren unterschiedlichen Regressionsmodellen bestimmen, wie aus dem demodulierten Interferogramm eine metrologisch verwertbare Längeninformation ermittelt wird. Das Interferogramm liegt nach Abschluss der Vorverarbeitung mit diskreten Stützstellen vor, an welche im Rahmen der Hauptverarbeitung das entsprechende Regressionsmodell angepasst wird. An der so ermittelten modellhaften Beschreibung des Interferogramms kann das Maximum des Interferenzkontrasts in Abhängigkeit vom gewählten Regressionsmodell zum Teil sub-nanometergenau bestimmt werden. Neben der metrologischen Güte der erzielbaren Ergebnisse wirkt sich die Wahl des Regressionsmodells jedoch auch wegen der zu berechnenden Parameter für alle Freiheitsgrade stark auf die erforderliche Verarbeitungsdauer zur Schätzung der Modellparameter aus.

Die optimale Wahl in Bezug auf die erzielte metrologische Güte der Messergebnisse ist die Nutzung einer Gaußverteilung als Hüllkurve, wodurch jedoch die Datenverarbeitung beachtlich verlangsamt wird. Die Dauer für die Messdatenverarbeitung liegt bei knapp über 12 Sekunden, was etwa der 4- bis 5-fachen der Dauer des Positioniervorgangs für eine Messposition entspricht und einen datenverarbeitungstechnischen Engpass erzeugt. Ein guter Kompromiss zwischen Datendurchsatz und Güte der erzeugten Daten ermöglicht die Modellierung der Einhüllenden durch eine Parabel. Die Verarbeitungsdauer von 2,9 Sekunden ist bei großflächigen Messungen[7] mit der Positionierdauer vergleichbar, wodurch bei Ausnutzung der Positionierzeiten für die WLI-Hauptdatenverarbeitung keine nennenswerten Wartezeiten zwischen der Rohdatenaufzeichnung benachbarter Positionen auftreten.

[7] Unter Annahme einer Positioniergeschwindigkeit ≤ 1 mm/s und einem Positionierweg zwischen 1 bis 0,1 mm.

6.4 Untersuchung des WLI-Parameterschätzverfahrens

Die Basis für die korrekte Funktion der WLI-Hauptverarbeitung ist die hinreichend exakte Modellierung des verwendeten Lichtquellenspektrums aus dominanter Wellenlänge und Kohärenzlängen, welche vom WLI-Parameterschätzverfahren automatisch aus den gewonnenen Rohdaten ermittelt werden. Die Basis für die Validierung des Parameterschätzverfahrens wurde wegen der Realitätsnähe ebenfalls auf dem Zielsystem (siehe Tabelle 18) durchgeführt. Die Eigenschaften des Parameterschätzverfahrens wurden mit einer Versuchsreihe aus 18 Einzelmessungen ermittelt. Als Lichtquelle diente eine 100 W Halogenleuchte. Für die Validierung wurde das Spektrum der Lichtquelle in eingebranntem Zustand mit einem Monochromator aufgezeichnet und mit dem neu entwickelten WLI-Parameterschätzverfahren ermittelt. Um eine Verfälschung des Emissionsspektrums durch das Remissionsspektrum des Messobjekts zu minimieren, wurde als Messobjekt ein Spiegel verwendet. Die Durchführung der WLI-Messungen erfolgte an 18 verschiedenen Regionen der Spiegeloberfläche. Die Schrittweite zwischen den Einzelaufnahmen betrug $\Delta l = 72$ nm.

Der in Kapitel 5.1.2.3 vorgestellte Schätzalgorithmus benötigt für die Auswertung von Interferogrammen mit 128 Stützstellen (0,721 ± 0,052) Sekunden. Im Rahmen der Schätzung wurden 1.000 gleichmäßig über den Bildbereich verteilte Interferogramme analysiert. Diese Anzahl ist für eine stabile Parameterschätzung hinreichend. Durch eine Vergrößerung der verwendeten Datenbasis wächst zwar die Verarbeitungsdauer linear, jedoch kann damit keine signifikante Steigerung der Schätzgüte erzielt werden. Die Schätzergebnisse der Messreihe sind in Tabelle 20 dargestellt.

Schätzverfahren	λ_0 [nm]	l_c [nm]	Verarbeitungs-dauer [s]
im Fourierraum	602,71 ± 1,67	1682,66 ± 31,17	0,721 ± 0,052
im Ortsbereich	603,67 ± 1,80	1669,03 ± 5,88	1,200 ± 0,094
Regression am Spektrum	601,11	1699,89	-

Tabelle 20: Ergebnisse der WLI-Parameterschätzung

Es ist zu erkennen, dass die beiden Modellparameter hinreichend genau aus den WLI-Daten geschätzt werden können. Die dominante Wellenlänge wird von beiden Verfahren treffsicher ermittelt, da dieser Parameter von beiden Verfahren mit vergleichbaren Algorithmen bestimmt wird. Der unterschiedliche Ansatz tritt insbesondere bei der Kohärenzlänge hervor. Zunächst kann die Vermutung bestätigt werden, dass die Berechnung im Ortsbereich nahezu eine Verdopplung der Verarbeitungsdauer nach sich zieht. Ferner zeigt sich, dass dieses Verfahren zwar eine deutlich schärfere Bestimmung der Kohärenzlänge ermöglicht, den Parameter aber um circa 30 nm zu klein schätzt. Auch die vergleichsweise große Unsicherheit des Schätzverfahrens im Fourierraum kann durch die Abweichung von Modell und Daten erklärt werden. Die Fourierkoeffizienten der Spektren streuen aufgrund von Rauschen im Bildaufnahmeprozess spürbar, was sich insbesondere in der Bestimmung der Signalbandbreite und damit der Kohärenzlänge niederschlägt.

Die fehlerbehafteten Kohärenzlängen beider Verfahren weisen jedoch bezogen auf den absoluten Wert geringe Abweichungen auf, welche bei der Weiterverarbeitung mit dem Matched-Filter vernachlässigbar sind. Ein Schätzfehler dieses Parameters bewirkt lediglich eine Skalierung der Filterantwort des Matched-Filters. Für die zu berechnende Distanz zur Objektoberfläche ist die absolute Amplitude jedoch nicht relevant.

6.5 Laufzeitverhalten der OSIS-Erweiterung

Um Schwachstellen in der Implementierung der OSIS-Schnittstelle aufzuzeigen ist das Wissen über das zu Grunde gelegte CORBA-Framework sehr hilfreich, da das schlanke OSIS-Framework selbst keine rechenintensiven Operationen ausführt, sondern nur ein Grundgerüst für Datentransport und Funktionensaufruf bereitstellt. Die vom Framework ermöglichten Funktionsaufrufe dienen letztendlich immer dem Aufruf von eingebetteten Objekten z. B. für die Datenerfassung von der Kamera oder die WLI-Datenverarbeitung.

Das Wissen um die Funktionsweise von CORBA lenkt somit bei der Suche nach potentiellen Schwachstellen das Hauptaugenmerk auf die Kommunikation zwischen den Framework-Klassen. Da Methodenaufrufe an entfernten Objekten nicht direkt möglich sind, werden die Aufrufe indiziert und mit den Aufrufparametern zu einem Befehlspaket zusammengefasst. Diese Pakete werden über das Netzwerk versendet und vom entfernten Objekt für den angeforderten Befehlsaufruf verwendet. Da ein Netzwerkzugriff generell langsamer ist als ein lokaler Aufruf, ist es sehr wahrscheinlich, dass die Datenübertragung zwischen den Komponenten *OSIS-Erweiterung* und der *dezentralen Sensorsteuerung* eine Engstelle bildet.

Eine Untersuchung der Datenübertragung am OSIS-konformen WLI-Sensorsystem bestätigt diese Vermutung. Der von OSIS vorgegebene Ablauf fordert die Übertragung aller gemessenen Rohdaten an die *OSIS-Erweiterung*, wofür die Datenaustauschmethode wiederholt mit je einem Messpunkt aufgerufen werden muss. Ein einzelner Aufruf überträgt lediglich 24 Byte Nutzdaten, wobei das übertragene Aufrufpaket insgesamt 212 Byte umfasst und weitere 26 Byte für die Quittierung des Aufrufs hinzukommen. Da der Funktionsaufruf für jeden Bildpunkt gesondert erfolgt, liegt die Datenmenge für die Kommunikationsorganisation fast eine Größenordnung über dem eigentlichen Nutzdatenaufkommen. Die OSIS-konforme Datenübertragung ist damit ineffizient und benötigt deshalb für die Übertragung von 307200 (640 mal 480) Punkten knapp eine Minute (siehe Tabelle 21).

Anzahl der Messpunkte	Datentransport OSIS-konform [s]	Datentransport zeilenweise [s]
307200	51,73 ± 0,31	0,34 ± 0,01

Tabelle 21: Dauer für den Datentransport zwischen den OSIS-Modulen

Um diesen Engpass zu beseitigen, wurde eine zusätzliche Methode für eine zeilenweise paketierte Übertragung der ermittelten 2,5D-Höhenkarte hinzugefügt. Da der Overhead der CORBA-Aufrufe proportional zur Anzahl der Aufrufe ist, kann das Nutzdaten-Gesamtlast-Verhältnis durch die paketierte Übertragung deutlich verbessert werden. So ist die Übertragung der gleichen Nutzdatenmenge in weniger als einer Sekunde realisierbar.

Die Übertragung der Höhenkarte im Ganzen könnte aufgrund des weiter reduzierten Overheads in noch kürzerer Zeit erfolgen. Davon wurde jedoch abgesehen, da bei gängigen Kameraauflösungen mehrere Megabyte Daten übertragen werden müssen, wodurch das Netzwerk kurzzeitig blockiert werden könnte. Das verwendete CORBA-Framework definiert zur Begrenzung der Paketgröße in der System-Registry den Wert *giopMaxMsgSize*, welcher die maximale Größe standardmäßig auf 2 Megabyte einschränkt. Es ist zwar möglich, diesen Grenzwert zu vergrößern, wovon aber im Hinblick auf Stabilität und Erreichbarkeit des Netzwerks abgesehen wurde.

6.6 Metrologischer Funktionsnachweis der NPMM-Steuerung

Im Folgenden sollen die implementierten Softwarekomponenten einem Integrationstest unterzogen werden. Das Ziel der Messungen ist eine globale Funktionsvalidierung und der Nachweis des korrekten Zusammenspiels der zentralen Komponenten der NPMM-Steuerung. Als Testobjekt für die Messungen wird ein Mikrokonturnormal der Physikalisch Technischen Bundesanstalt verwendet [124]. Die Messung am PTB Mikrokonturnormal erfolgt in der Region „area4“ (siehe Abbildung 69). Der Bereich besteht aus jeweils drei quaderförmigen Erhebungen und Vertiefungen mit Kantenlängen von 1 mm, 0,5 mm und 0,2 mm. Zur Messung wurde ein Silizium-Membran-Sensor mit 0,3 mm Rubinkugeltaster verwendet [55].

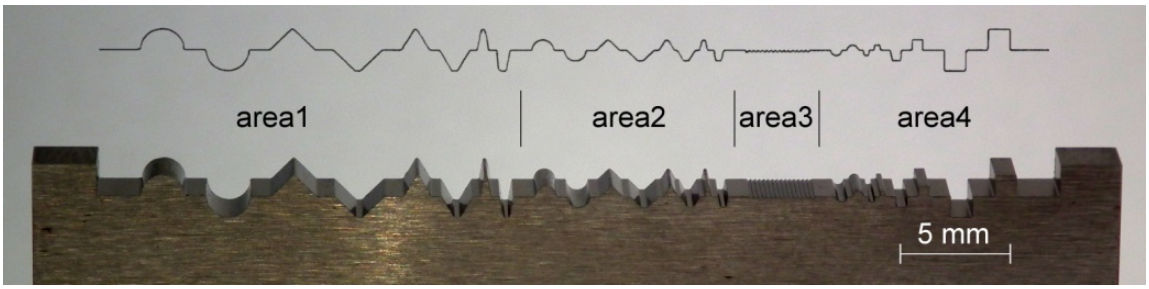


Abbildung 69: Gesamtansicht des PTB Mikrokontur-Normals

Die zu vermessenden Punkte sind in Abbildung 70 dargestellt. Die Antastung der Messpunkte erfolgt auf der Stirnfläche des Konturnormals 1 mm entfernt von der Vorderkante.

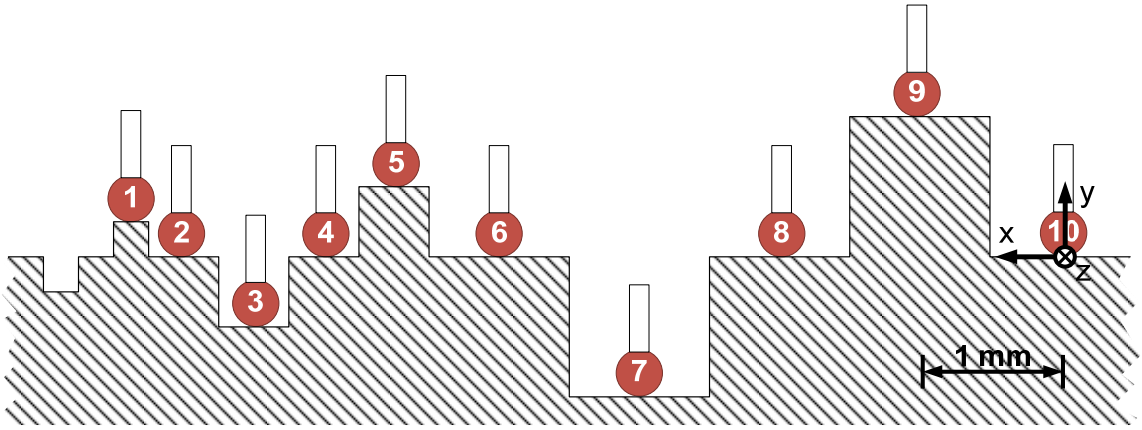


Abbildung 70: Nominelle Antastorte der Stufenhöhen am PTB Mikrokontur-Normal

6.6.1 Messungen mit taktilen Sensoren

Da die Ausrichtung des Objektes im Messvolumen der NPM-Maschine a priori nicht exakt bekannt ist, muss vor der eigentlichen Messung ein am Mikrokonturnormal fest ausgerichtetes Bauteilkoordinatensystem eingemessen werden. Dazu werden drei orthogonale Flächen an hinreichend vielen, möglichst gleichmäßig über eine Objektfläche verteilten Messpunkten angetastet. Mit der Auswertungssoftware Quindos7 können anhand der gemessenen Punkte Ausgleichsebenen ermittelt werden, womit wiederum die Lage und Orientierung des Bauteilekoordinatensystems festgelegt werden kann. Die Ausrichtung des Koordinatensystems erfolgt so, dass sich der Ursprung im Messpunkt 10 befindet und die Raumachsen entlang der dominanten Objektkanten verlaufen. Die Antastung der Messpunkte kann unter Nutzung des Teilkoordinatensystems und einer Beschreibung des Normals ohne komplizierte Transformationsberechnungen sehr einfach in Quindos 7 programmiert werden.

Die Messsequenz wird 16 Mal wiederholt. Aus den gemessenen Rohdaten werden die mittlere Höhenausdehnung der Strukturen und die Messunsicherheit ermittelt. Die Ergebnisse sind grafisch in Abbildung 71 und numerisch in Tabelle 22 präsentiert.

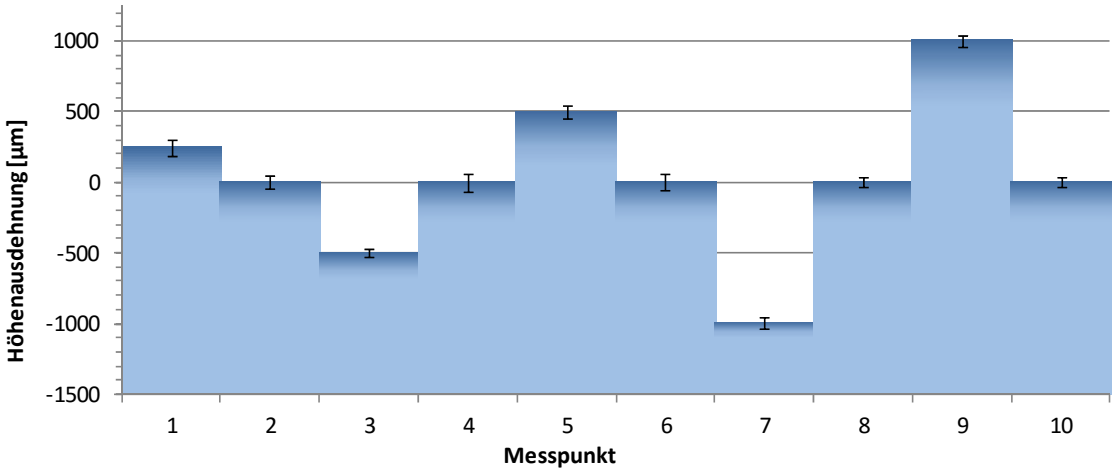


Abbildung 71: Darstellung der taktil gemessenen Stufenhöhen
(Die angegebenen Messfehlerintervalle sind um den Faktor 10000 vergrößert dargestellt.)

Stufe	Stufenhöhe [µm]		Differenz [nm]
	gemessen	Kalibrierschein	$\Delta l_{Messung,\ Schein}$
1	246,712 ± 0,015	247 ± 0,9	288 ± 885
3	497,157 ± 0,014	498 ± 0,6	843 ± 586
5	499,163 ± 0,017	499 ± 0,5	163 ± 483
7	996,733 ± 0,014	998 ± 1,5	1267 ± 1486
9	999,832 ± 0,011	999 ± 0,8	832 ± 789

Tabelle 22: Taktil gemessene Stufenhöhe verglichen mit den Kalibrierdaten der PTB

Die Messunsicherheit für die durchgeführte Antastserie bewegt sich in Abhängigkeit vom Messpunkt zwischen 11 nm bis 17 nm. Der Vergleich der Messdaten in Tabelle 22 mit den Kalibrierdaten der PTB zeigt unter Berücksichtigung der im Kalibrierschein angegebenen Oberflächenrauheit (*Rz* = 1,8 µm und *Ra* = 0,17 µm) eine für die Validierung hinreichende Übereinstimmung der ermittelten Stufenhöhen mit den Daten der PTB, wodurch die Korrektheit der internen Datenverarbeitung und ihre Tauglichkeit für metrologische Anwendungen innerhalb der Unsicherheitsgrenzen des PTB-Normals bestätigt wird. Für etwaige weiterführende Untersuchungen sind genauere Normale mit Unsicherheiten im (Sub-)Nanometerbereich erforderlich.

6.6.2 Messungen mit OSIS-konformen optischen Sensoren

Zur Validierung der WLI-Sensor Implementierung wird wegen der Vergleichbarkeit der Ergebnisse ebenfalls das PTB Mikrokonturnormal verwendet. Die Lage der nominellen Antastpunkte ist mit denen der taktilen Messung identisch. Jedoch ist es aufgrund der physikalischen Beschränkungen der Weißlichtinterferometrie nicht möglich, das Messobjekt automatisiert in der oben beschriebenen Weise auszurichten.

Für die automatisierte Ausrichtung müssen die drei dominanten Ebenen des Messobjekts bestimmt werden. Diese sind jedoch orthogonal zueinander und reflektieren nicht die hinreichende Lichtmenge, um messbare Interferenzen zu erzeugen. Folglich ist es erforderlich, das Messobjekt manuell auszurichten, weshalb die Antastpunkte der optischen Messung nicht notwendigerweise mit denen der taktilen Messung übereinstimmen. Da die Rohdatenaufzeichnung bei der optischen Antastung per

WLI aufgrund der vertikalen Scanbewegung und der komplexen Auswertungsverfahren deutlich länger dauert als bei einer taktilen Antastung, wurde die Anzahl der Wiederholungen auf 10 reduziert, um die thermische Drift während der Messung in Grenzen zu halten. Aus den gemessenen Rohdaten wurden die mittleren Höhenausdehnungen der Strukturen und die Messunsicherheit ermittelt. Die Ergebnisse sind grafisch in Abbildung 72 und numerisch in Tabelle 23 präsentiert.

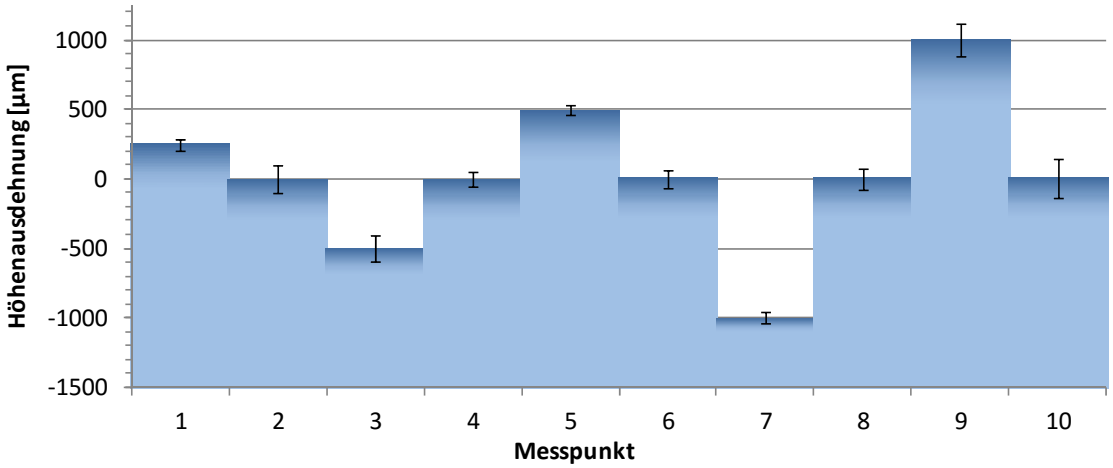


Abbildung 72: Darstellung der optisch gemessenen Stufenhöhen
(Die angegebenen Messfehlerintervalle sind um den Faktor 10.000 vergrößert dargestellt.)

Messpunkt	Stufenhöhe [µm]		Differenz [nm]	
	gemessen	Kalibrierschein	$\Delta l_{Messung,\ Schein}$	$\Delta l_{optisch,\ taktil}$
1	245,640 ± 0,108	247 ± 0,9	1360 ± 792	1072 ± 93
3	497,542 ± 0,120	498 ± 0,6	458 ± 480	385 ± 106
5	499,273 ± 0,078	499 ± 0,5	273 ± 422	110 ± 61
7	998,236 ± 0,094	998 ± 1,5	236± 1406	497 ± 80
9	999,332 ± 0,167	999 ± 0,8	332 ± 633	500 ± 156

Tabelle 23: Optisch gemessene Stufenhöhe verglichen mit den Kalibrierdaten der PTB

Die Messunsicherheit für die durchgeführte Antastserie bewegt sich in Abhängigkeit vom Messpunkt zwischen 78 nm und 167 nm. Ein Vergleich der Messdaten in Tabelle 23 mit den Kalibrierdaten der PTB zeigt unter Berücksichtigung der Oberflächenrauheit und der Messunsicherheit eine hinreichende Übereinstimmung der ermittelten Stufenhöhen mit den Kalibrierwerten der PTB und bestätigt damit die korrekte Verarbeitung von Messdaten sowohl durch die OSIS-Komponenten als auch durch die Steuerungssoftware. Das OSIS-Framework und die darauf aufbauende Sensorimplementierung für WLI wurden somit unter Berücksichtigung der Unsicherheit des PTB-Normals erfolgreich validiert.

6.7 Vergleich von taktiler Einzelpunktmessung und Scanmessung

Zum Vergleich der Antaststrategien Einzelpunktmessung und Scanmessung wurden weitere Untersuchungen angestellt. Dazu war es jedoch erforderlich, die Messgeschwindigkeit für die Scanmessung festzugelegen. Da die Messdauer direkt von der Geschwindigkeit abhängt, sollte diese möglichst hoch gewählt werden, um Umwelteinflüsse auf die Messung gering zu halten. Hohe Messgeschwindigkeiten wirken sich unter gewissen Umständen jedoch auch negativ auf den Messprozess aus. Bei Scanmessungen kann es durch Resonanzeffekte aufgrund des kontinuierlichen Kontakts mit der Objektoberfläche oder der Trägheit der Antastkraftregelung zu Messfehlern kommen, welche bei der Einzelpunktantastung nicht auftreten.

6.7.1 Festlegen der Scangeschwindigkeit

Um mit dem Positioniersystem NMM-1 Scanmessungen ausführen zu können, ist es zunächst erforderlich, einen geeigneten Parametersatz für die Antastkraftregelung festzulegen. Generell muss davon ausgegangen werden, dass der PI-Regler nicht bei allen Parameterkombinationen stabil arbeitet, wodurch die Antastregelung entweder zu träge ist und der Objektoberfläche nicht folgen kann oder der Regler zu stark reagiert und es zum Aufschwingen und infolgedessen zur massiven Störung der Messdaten kommen kann. Die Notwendigkeit einer geeigneten Parametrierung zeigte sich bereits bei einem ersten Testscan mit einer Scangeschwindigkeit von $v_{scan} = 1\ \mu m/s$. Mit der standardmäßig eingestellten Verstärkung des PI-Reglers von $k_n = 1$ ist die Antastregelung nicht in der Lage, der Topographie des Messobjekts zu folgen. Während der Scanmessung verliert die Regelung den Kontakt zur Oberfläche bzw. stoppt die Kraftüberwachung die Scanbewegung wegen des Überschreitens der maximal erlaubten Tasterauslenkung. Durch ein Erhöhen des Verstärkungsparameters kann die Dynamik des Reglers erhöht werden, wodurch dieser auch bei hohen Scangeschwindigkeiten in der Lage ist, der Oberflächentopographie zu folgen.

Um mögliche Einflüsse der Parametrierung auf die Messdaten zu untersuchen, wurde ein Stahldorn des in Abbildung 73 dargestellten Messobjekts mit verschiedenen Scangeschwindigkeiten zwischen 10 µm/s und 100 µm/s bei unterschiedlicher Parametrierung des Antastreglers entlang der selben Trajektorie gescannt. Für jeden Parametersatz wurden 20 Wiederholungsmessungen ausgeführt, für welche die in Tabelle 24 wiedergegebenen Formabweichungen ermittelt wurden.

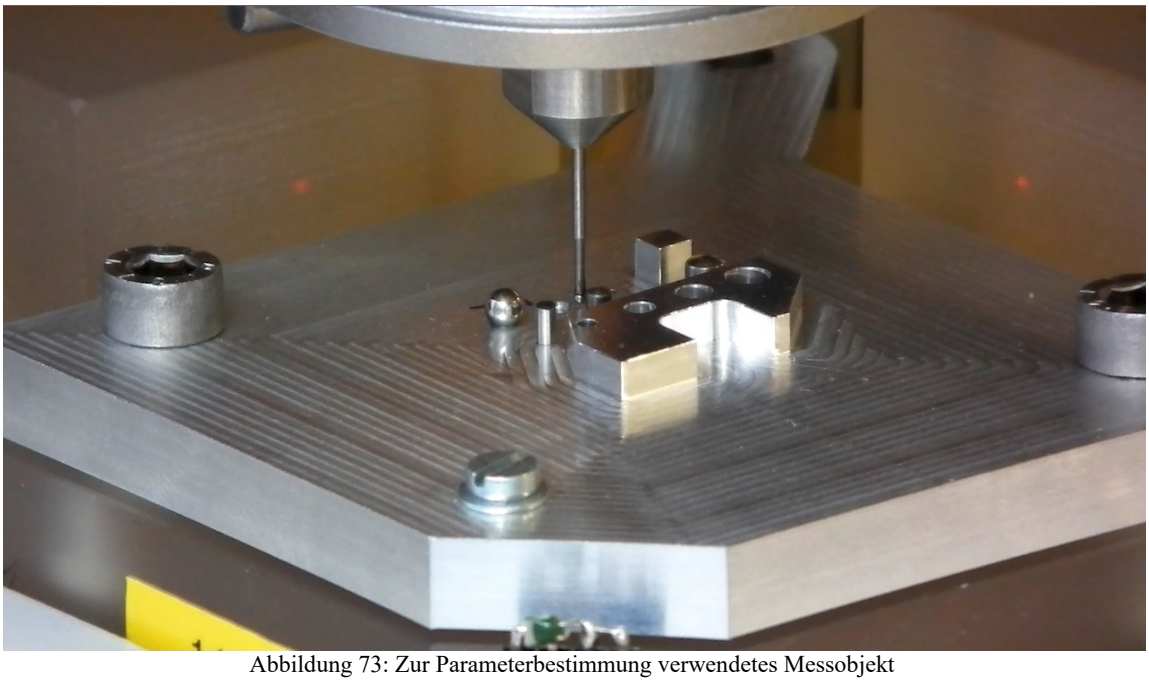


Abbildung 73: Zur Parameterbestimmung verwendetes Messobjekt

v [µm/s] / K_n [-]	100	50	30	10
33	-	-	-	8,2 nm
100	-	-	7,6 nm	8,2 nm
300	-	8,5 nm	7,4 nm	8,0 nm
500	7,6 nm	7,3 nm	7,04 nm	7,8 nm

Tabelle 24: Mittlere Formabweichung in Abhängigkeit der Scanparameter

Insbesondere zeigte sich, dass ein proportionaler Zusammenhang zwischen der gewählten Scangeschwindigkeit und der dafür erforderlichen Reglerverstärkung k_n besteht. Bei hohen Scangeschwindigkeiten muss der Regler wesentlich dynamischer agieren, um der Topographie folgen zu können, da die Messung anderenfalls wegen einer Überschreitung der maximal zulässigen Tasterauslenkung von der Sensorüberwachung gestoppt wird. Die Ergebnisse der Messungen mit einer hinreichend agilen Reglerparametrierung weisen hingegen keinen signifikanten Zusammenhang bezüglich der Scanparameter auf. Die Unsicherheit liegt bei allen Konfigurationen zwischen 7 nm und 8,5 nm. Die geringe Streuung legt nahe, dass bei den untersuchten Scanparametern auf der Oberfläche dieses Messobjekts keine durch Aufschwingen verursachten Fehler auftreten.

6.7.2 Analyse der Vergleichsmessung

Basierend auf den vorigen Untersuchungen und dem Ziel die Messdauer möglichst gering zu halten, werden die Scanparameter für die Vergleichsmessung damit auf die Parameter $v_{scan} = 100$ µm/s und $k_n = 500$ festgelegt. Für den Einzelpunkt-Messmodus sind weitere Geschwindigkeiten für das Positionieren, das Annähern an bzw. Entfernen vom Objekt zu definieren. Da der Taster in diesen Phasen mit dem Objekt nicht in Kontakt kommen kann, wird die Geschwindigkeit auf $v = 1$ mm/s festgelegt. Für den eigentlichen Antastvorgang wird die Geschwindigkeit auf $v_{scan} = 100$ µm/s reduziert.

Da insbesondere die Messung im Einzelpunktmodus eine sehr lange Messdauer erfordert, konnten Störungen der Messung durch Vibrationen nicht gänzlich vermieden werden, welche sich als Ausreißer in den Messdaten niederschlagen (siehe dritter Quadrant in Abbildung 74) und bei der Messdatenauswertung nicht berücksichtigt wurden.

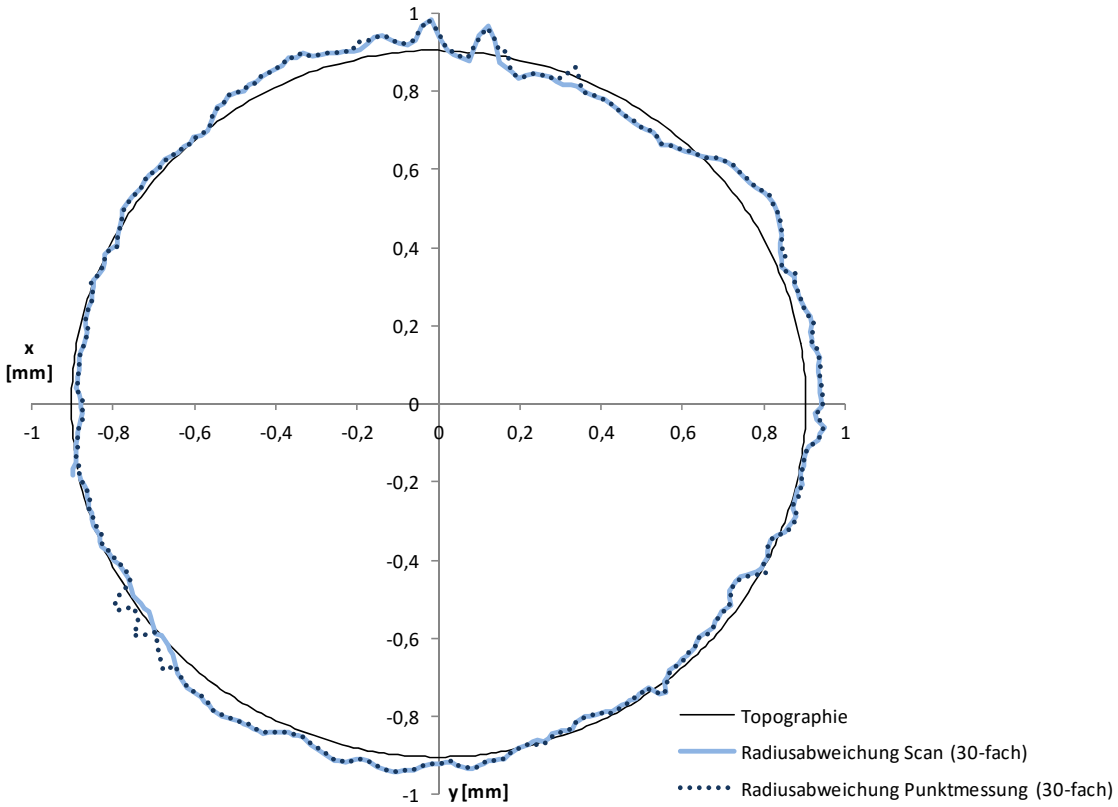


Abbildung 74: Darstellung der Vergleichsmessung Punktantastung und Scanmessung

Der Vergleich weist eine gute Übereinstimmung der Ergebnisse auf. Die weiterführende Analyse der Daten (siehe Tabelle 25) zeigt, dass die gefitteten Kreismittelpunkte einen Unterschied von 92 nm und 28 nm in x- bzw. y-Richtung aufweisen und der mittlere Radius der Scanmessung um 89 nm kleiner ist als bei der Einzelpunktantastung. Die Ursache hierfür kann in thermischer Drift und durch das Gebäude übertragene Erschütterungen des Messaufbaus vermutet werden.

	Punktmessung	**Scanmessung**	**Differenz**
Messdauer [min]	25	1	23
Radius [µm]	903,238	903,149	0,089
Rundheitstoleranz [µm]	4, 402	4,114	0,288

Tabelle 25: Mit der Vergleichsmessung ermittelte Parameter

6.8 Validierung der optischen Übersichtssensoren

6.8.1 2D-Übersichtssensor

Die Validierung der Softwarekomponenten des 2D-Übersichtssensors erfolgt wie bei der NPMM-Steuerung mit experimenteller Messung, welche alle relevanten Anwendungsfälle umfasst. Dementsprechend muss der Sensor zunächst mit einem geeigneten Gitternormal kalibriert werden (siehe Abbildung 75), bevor die eigentliche Aufzeichnung der Rohdaten und die finale Erzeugung der 2D-Übersicht des Messvolumens validiert werden kann.

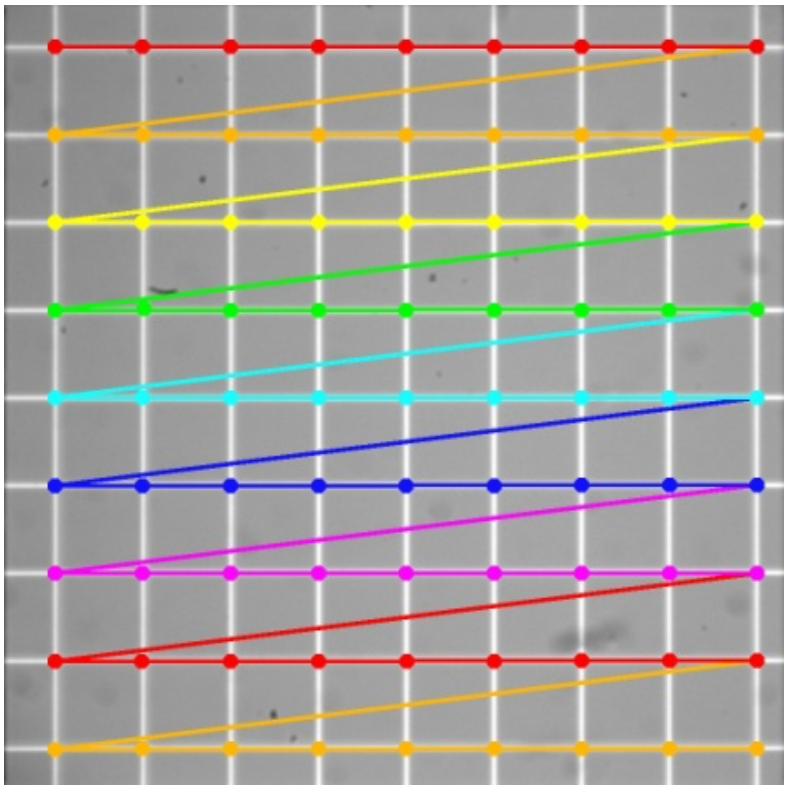


Abbildung 75: Gefundene Messpunkte bei der 2D-Kalibrierung an einem Gitternormal

Die mit der 2D-Kalibrierung ermittelten Verzeichnungsparameter (siehe Tabelle 26 und Tabelle 27) zeigen auf, dass die Objektive verzeichnungsarm sind. Die Gesamtverzeichnung wird vom tangentialen Modellanteil dominiert (siehe Abbildung 76), was auf eine Verkippung des Kamerasensors relativ zur Achse des optischen Systems schließen lässt. Der Restfehler der Kalibrierung (siehe Tabelle 27) beträgt $U = 1{,}31 \cdot 10^{-4}$, was sich über der Gesamtausdehnung der 2D-Übersicht zu einem maximalen Fehler von $\Delta p = 4{,}42$ µm akkumuliert.

Parameter	Wert
Verzeichungsmodell radiale Parameter (k1, k2, k3)	-9,31·10-6 -1,05·10-9 -5,53·10-13
Verzeichungsmodell tangentiale Parameter (p1, p2)	-1,73·10-2 -1,32·10-1
maximale intrinsische Verzeichnung (korrigierbar)	5,93 Pixel (2,48 µm)

Tabelle 26: Intrinsische Parameter

Parameter	Wert
Rotationswinkel [°]	179,83
laterale Auflösung [nm]	417,41
Kalibrierrestfehler	≤ 1,31·10-4 (131 nm je 1 mm)

Tabelle 27: Extrinsische Parameter

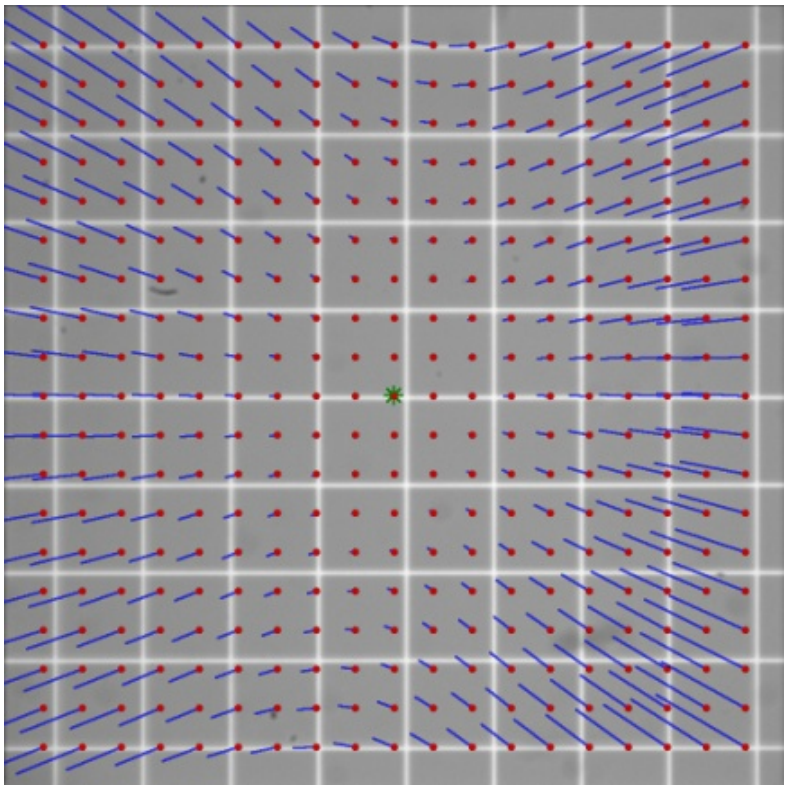


Abbildung 76: Verzeichnung des optischen Systems
(um den Faktor 100 vergrößert dargestellt)

Durch die nachträgliche Korrektur dieses Fehlers kann der statische Messfehler insbesondere in den Randbereichen der Einzelaufnahmen verringert werden, wodurch beim Zusammenfügen Kanten bzw. Sprünge an den Rändern benachbarter Bildbereiche vermieden werden können. Nach Abschluss der 2D-Kalibrierung sind alle Parameter für die Transformation von Punkten aus dem Koordinatensystem des Positioniersystems in das Sensorkoordinatensystem bekannt und das Messvolumen kann durch ein entsprechendes Positionierregime erfasst und diese Daten zu einer 2D-Übersicht gestitcht werden. Das Messvolumen wurde mit 7227 Einzelaufnahmen abgebildet (siehe Tabelle 28) und diese zu einer Gesamtansicht in Abbildung 77 zusammengesetzt.

Parameter	Wert
Grauwerttiefe [Bit]	8
Einzelbildauflösung [Pixel]	582 x 782
Anzahl Einzelbilder	99 x 73
Übersichtsbereich [mm]	23,896 x 23,828
Messdauer [h]	2,21
Dateigröße [GByte]	3,063

Tabelle 28: Eigenschaften der gestitchten 2D-Übersicht

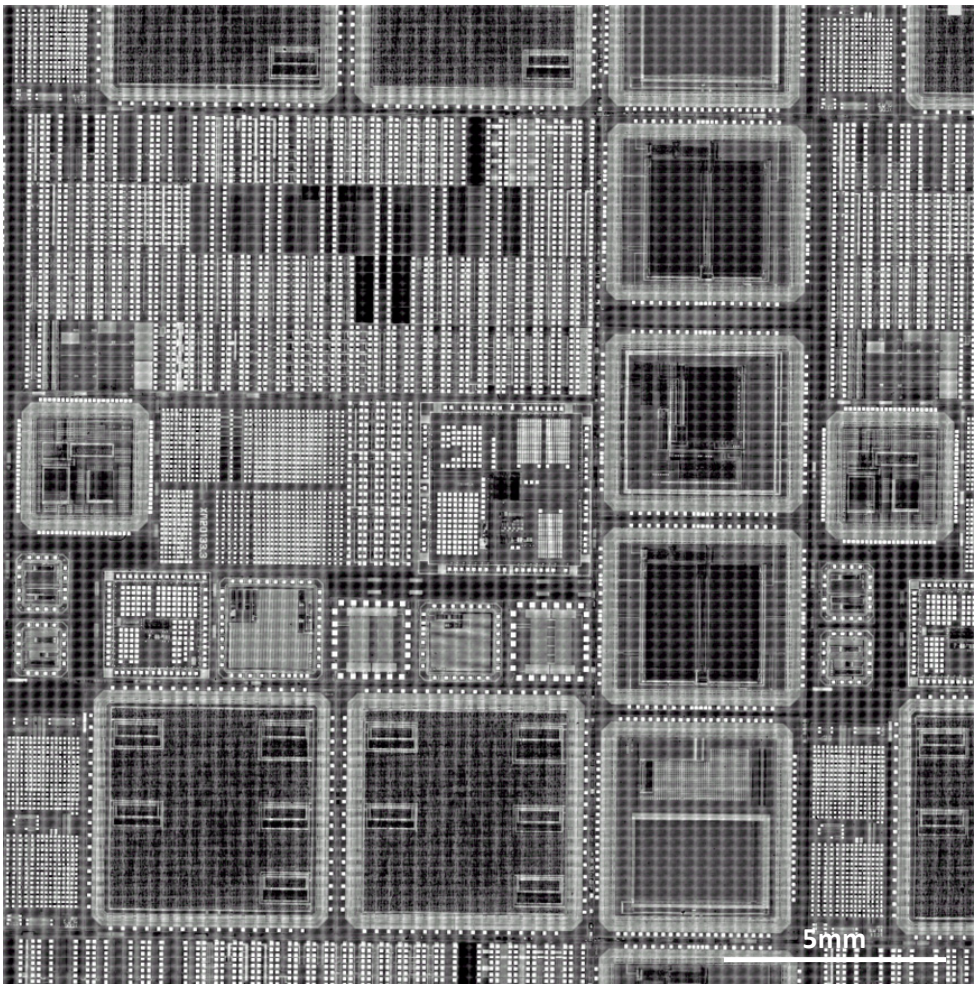


Abbildung 77: Gestitchte 2D-Messvolumenübersicht mit Halbleiterstrukturen

Das erzeugte Übersichtsbild visualisiert den Messbereich und erlaubt die detaillierte Analyse lokaler Mikrostrukturen. Die Messvolumenübersicht kann den Aufbau eines kompletten Prozessorkerns abbilden und stellt lokale Strukturen bis zur Grenze der optischen Auflösung dar. Die Übersicht ermöglicht damit die manuelle Inspektion (z. B. durch Zoomen) des Objekts bzw. die automatisierte Detektion interessanter Bereiche, um in diesen Regionen weitere Untersuchungen mit höher auflösenden Sensoren auszuführen.

6.8.2 3D-Übersichtssensor

6.8.2.1 Güte der erfassten 3D-Daten

Um das Rauschen des Stereovisionsystems zu beurteilen, wurde eine Fläche mit der Kantenlänge von 200 x 200 mm² (Ausdehnung in xy-Richtung) ohne ein Messobjekt erfasst. Auf den texturlosen, ebenen Untergrund wurde mit einem Miniprojektor ein zufälliges Schwarz-Weißmuster projiziert und diese Textur für die Suche von korrespondierenden 3D-Punkten verwendet. Für die Auswertung wurde eine Ausgleichsebene in die gemessene 3D-Punktewolke eingepasst.

Als Basislänge des Stereosystems wurde mit der Kalibrierung ein Kameraabstand von 48,83 mm ermittelt. Bei den 15 Messungen wurde zwischen 9216 und 9188 (im Mittel 9203) Punkte in der Ebene erfasst. Die Anzahl schwankt, da aufgrund von Bildrauschen nicht alle korrespondierenden Stereopaare zugeordnet werden können. Die statistische Auswertung der Messreihe ist in Tabelle 29 wiedergegeben.

Messreihe	Streubreite [mm]	Standard-abweichung [µm]	Unsicherheit des Ausgleichs [µm]
1	5,256	660,382	6,887
2	5,122	661,515	6,899
3	5,206	657,052	6,853
4	5,243	661,235	6,894
5	5,227	663,610	6,920
6	5,304	661,434	6,898
7	5,381	659,220	6,869
8	5,146	659,477	6,880
9	5,228	656,668	6,843
10	5,056	651,977	6,794
11	4,936	656,415	6,841
12	5,071	656,147	6,837
13	5,089	658,474	6,859
14	5,267	656,941	6,846
15	5,266	659,239	6,867

Tabelle 29: Messunsicherheitsparameter des Stereosystems

Im Mittel ist die gefittete Ausgleichsebene mit einer Unsicherheit von 6,9 µm behaftet, die Standardabweichung der 3D-Punktewolke beträgt 658 µm und die absolute Streubreite um die Gaußebene beträgt im Mittel 5187 µm. Die Daten belegen, dass die Messunsicherheit des Stereovisionsystems für den beabsichtigten Einsatz als System zur Gewinnung von Übersichtsdaten für die Planung der Prüfstrategie geeignet ist. Die erzielte Standardabweichung der Daten liegt nur geringfügig über dem doppelten Durchmesser der Antastkugel des verwendeten Mikrotasters.

Eine noch ungelöste Fragestellung ist allerdings die Detektion und Beseitigung von Ausreißern. Das mit 5,2 mm sehr breite Toleranzband ist für die Prüfplanung an Mikrostrukturen zu groß, um unmittelbar auf den Rohdaten operieren zu können. Da Messobjekte in der Regel jedoch segmentweise stetig verlaufen und keine nadelartigen Strukturen mit einer Ausdehnung von mehreren Millimetern besitzen, ist es möglich, die Ausreißer durch adaptive Filter oder lokale Ausgleichsalgorithmen zu eliminieren.

6.8.2.2 3D-Reproduzierbarkeit

Die Beurteilung der 3D-Reproduzierbarkeit erfolgte durch die experimentelle Vermessung eines synthetischen Referenzobjekts. Dieses besteht aus zwei Positionsmarken mit dem definierten Abstand von d = 80 mm. Es wurden vier Messreihen aufgenommen, wobei das Referenzobjekt jeweils koaxial zu den drei Raumachsen des Weltkoordinatensystems und einmal windschief im Messvolumen ausgerichtet wurde. Die Ergebnisse sind in Tabelle 30 zusammengefasst. Obwohl der globale, über alle Messreihen bestimmte, mittlere Abstand der Positionsmarken mit d = 79,88 ± 0,082 mm gut mit dem Referenzobjekt überein stimmt, zeigen sich jedoch achsspezifische Unterschiede.

Ausrichtung Referenzobjekt	mittlerer Abstand [mm]	Standard-abweichung [mm]	Unsicherheit [mm]
x	79,809	0,315	0,113
y	80,027	0,461	0,166
z	80,242	0,305	0,110
xyz	80,409	0,545	0,196

Tabelle 30: 3D-Reproduzierbarkeit des Stereosystems

Bei achsparalleler Anordnung des Referenzobjekts wird der Abstand der Positionsmarken gut wiedergegeben. Die Abweichung liegt unter 250 µm. Die windschiefe Anordnung weist eine fast doppelt so große Abweichung auf. Unter Einbeziehung der experimentellen Bedingungen sind diese Daten jedoch zu erwarten, da die Tiefenauflösung des Systems keine genaueren Aussagen zulässt.

Die Tiefenauflösung des Stereosystems kann wie folgt bestimmt werden [107]:

$$\Delta z = \frac{{d_z}^2}{bf} d_{min} \tag{6.7}$$

Die Auflösung Δz ist von der minimal detektierbaren Disparität d_{min}, dem Quadrat der Entfernung d_z zwischen Sensor und Objekt und dem reziproken Produkt von Basisabstand b und Brennweite f abhängig.

Nach Williamson [125] ist für die kleinste detektierbare Disparität ein Wert von $\frac{1}{4}$ Pixel anzunehmen. Die übrigen experimentellen Parameter betrugen $z = 300\,\text{mm}$, Basisabstand und Brennweite wurden durch die Stereokalibrierung mit $b = 48{,}83\,\text{mm}$ und $f = 7{,}59\,\text{mm}$ ermittelt. Die am Arbeitspunkt erreichbare Auflösung beträgt damit nach (6.7) $\Delta z = 0{,}449\,\text{mm}$. Alle gemessenen Referenzmarkenabstände weichen weniger als die lokale Tiefenauflösung vom tatsächlichen Abstand der Referenzmarken ab. Das System arbeitet damit erwartungsgemäß und stellt die beobachtete Szene im Rahmen seiner Auflösungsgrenzen korrekt dar.

Kapitel 7 Untersuchung des Zusammenwirkens von NPM-Maschinen und I++DME Steuerungssoftware

Das abschließende Kapitel umfasst die Darstellung der im Rahmen der experimentellen Untersuchungen gesammelten Erfahrungen im Umgang mit I++DME konformer Messsoftware. Trotz ihrer separaten Entwicklung, der historisch bedingten Entstehung mit makroskopischen KMGs und der erst in jüngster Zeit integrierten I++DME Unterstützung sind die getesteten Mess- und Steuerungsanwendungen sowohl im Arbeitsablauf als auch in ihren Grundfunktionen nahezu identisch.

Die generelle Arbeitsweise der getesteten I++DME Messprogramme wird unter Abstraktion der spezifischen Steuerung im folgenden Abschnitt zusammengefasst. In den sich daran anschließenden Abschnitten werden die getesteten I++DME Clients darüber hinaus detaillierter erläutert und ihre spezifischen Fähigkeiten einander gegenübergestellt.

7.1 Arbeitsweise beim Messen mit einem I++DME Client

Die im experimentellen Teil der Arbeit angewandte Vorgehensweise zur Vorbereitung und Durchführung einer Messung folgt bei allen getesteten I++DME Clients einem stets gleichen Schema und kann in die folgenden sieben Schritte unterteilt werden:

- Laden der CAD-Daten des zu untersuchenden Objekts
- Festlegen eines Bauteilkoordinatensystems am CAD-Modell
- Definieren von Formelementen, Messpunkten und Fahrwegen
- Festlegen der auszuwertenden Parameter und ihrer Toleranzen
- Kalibrieren des Tasters
- Einmessen des Messobjekts
- Durchführung der Messung

Alle I++DME Messprogramme stellen die geladenen CAD-Daten, welche die Grundlage für das Festlegen der Antastpositionen bilden, grafisch dar und ermöglichen das interaktive Festlegen von zu vermessenden geometrischen Primitiven per Mausklick. Diese Primitiven sind entweder Punkte, Linien, Kreise, Ebenen, Zylinder oder je nach Unterstützungsumfang der Programme auch komplexere Elemente wie z. B. Tori. Diesen Formelementen werden anschließend mit der NPM-Maschine angetastete Messpunkte zugeordnet, anhand derer das Messprogramm nach Abschluss der Messung die Auswertung vornimmt. Dazu wird das Formelement durch Ausgleichsrechnung in die gemessenen Daten eingepasst und die Maßhaltigkeit des Objekts auf Basis der daran ermittelten Parameter bewertet.

Da sich die Arbeitspunkte des Tasters durch Temperaturschwankungen, Alterung oder Kollisionen verändern können, muss dieser vor jeder Messung an einem Normal kalibriert werden. Als Kalibrierkörper wird in der Regel eine präzise gefertigte und wohl bekannte Kugel verwendet, welche aus verschiedenen Richtungen mit Einzelpunktantastung (Quindos7) oder Scan-Messungen (Calypso 5.2) angetastet wird. In die gemessenen Punkte wird eine Kugel eingepasst und aus der Differenz zwischen dem bekannten Radius der Kalibrierkugel und dem Radius der Regression der effektive Tasterradius bestimmt. Der effektive Radius ist dabei nicht identisch mit dem Radius des Tastelements, vielmehr setzt er sich aus dem tatsächlichen Radius der Tastkugel und der Deflektion des Schafts aufgrund der Antastkraft zusammen.

Um die eigentliche Messung auszuführen, ist es unabdingbar, einen Bezug zwischen Messobjekt und KMG herzustellen. Diese Bezugnahme erfolgt durch die Bestimmung der affinen Transformation zwischen den intrinsischen Koordinatensystemen von Messobjekt und KMG. Dazu werden am Objekt drei geeignete, vorab definierte Ebenen manuell angetastet und die Transformation anhand ihrer Lage und Orientierung ermittelt. Anschließend kann die am CAD-Modell geplante Messaufgabe automatisiert ausgeführt werden.

Diese Verfahrensweise hat im Rahmen der makroskopischen KMT durchaus ihre Berechtigung, kann für die Nanomesstechnik jedoch nicht direkt angewendet werden. Im Bereich der Nanometerskala ist davon auszugehen, dass die relative Toleranz strukturierter Objekte deutlich größer als bei makroskopischen Objekten ausfällt. Dies hat zur Folge, dass ein auf die Antastung von sechs Punkten gestütztes Einmessen des Messobjekts einer stark erhöhten Unsicherheit unterliegt. Ein nanometrologisch geeigneter Ansatz wäre die Nutzung einer breiteren Datenbasis, um die Lage und Orientierung des Messobjekts zu bestimmen. Durch die Erfassung der Ebenen mit Scanmessungen und die anschließende Ermittlung der Transformation anhand der gemessenen Punktwolken bzw. darin

eingepasster Ausgleichsebenen wäre es möglich die Unsicherheit der Transformation deutlich zu senken.

7.2 Abgrenzung der Koordinatenmessprogramme zueinander

Die Inbetriebnahme der Steuerung für NPM-Maschinen erfolgte mit den drei Messanwendungen Quindos7 (Hexagon Metrology), Calypso DME (Zeiss Industrielle Messtechnik) und Metrolog (Renishaw). Diese Programme werden im Folgenden erläutert und auf ihre Vor- und Nachteile eingegangen.

7.2.1 Quindos7

Die Anfänge von Quindos liegen im Jahr 1982. Das Programm wurde kontinuierlich weiterentwickelt und um neue Funktionen ergänzt, wodurch es sehr umfangreich und vielschichtig wurde. Quindos7 besitzt durch die funktionelle Abwärtskompatibilität bis zur Urversion einen sehr großen Umfang an Messprozeduren und ermöglicht durch seine proprietäre Programmiersprache eine beliebig komplexe Verknüpfung von Algorithmen auch für hoch komplexe Messaufgaben. Dadurch ist Quindos7 den Messprogrammen Calypso und Metrolog funktionell deutlich überlegen, erfordert aber auch eine lange Einarbeitungszeit und viel Erfahrung in der Anwendung. Quindos7 ist damit eher ein Messprogramm für "Power-User" mit besonderem Fokus auf Flexibilität und hohe Funktionalität, was das Programm für die akademische Grundlagenforschung qualifiziert. Quindos7 stellt für die Festlegung der Messaufgabe und die Auswertung der gemessenen Daten eine proprietäre prozedurale Programmiersprache bereit, auf welcher alle Messprozeduren basieren (siehe Abbildung 78).

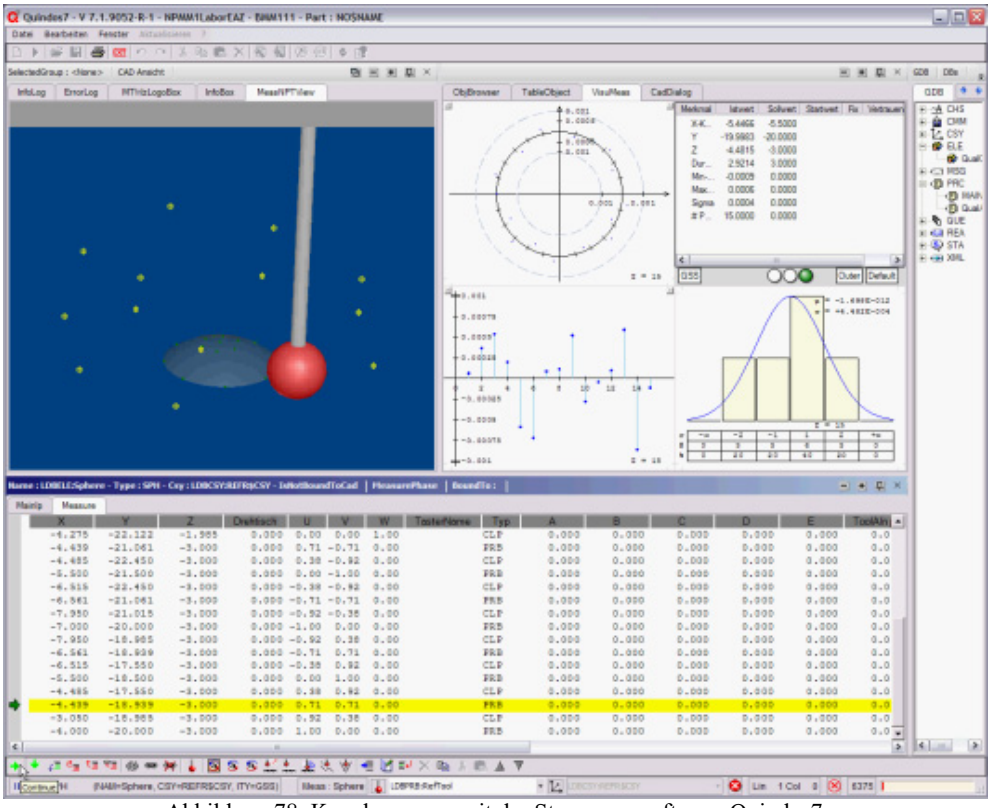


Abbildung 78: Kugelmessung mit der Steuerungssoftware Quindos7

Der Umfang der mitgelieferten Prozeduren ist sehr umfangreich und kann von erfahrenen Anwendern erweitert und flexibel an die eigenen Bedürfnisse angepasst werden. Darüber hinaus erlaubt es auch die Berechnung von abstrakten oder sehr komplexen Parametern. Für die Speicherung von Daten und

Zwischenergebnissen wird eine vielschichtige Datenbankstruktur verwendet, welche zwar die Einarbeitung erschwert, aber erfahrenen Nutzern eine hilfreiche Unterstützung bei der Programmierung bietet. Auch für die Tastkugelkalibrierung wird von Quindos7 eine Prozedur bereitgestellt, welche unter Angabe des Radius der verwendeten Kalibrierkugel und eines Punktes am Pol selbsttätig Antastpunkte für die Kalibrierung berechnet, anfährt und mit den gewonnenen Daten den effektiven Tastkugelradius berechnet, wobei für die Kalibrierung Einzelpunktmessungen genutzt werden.

Um Kollisionen zwischen Taster und Kalibrierkugel zu vermeiden, wird der Taster zwischen den Messungen auf entfernte Parkpositionen gefahren. Diese Parkpositionen sind mehrere Millimeter von der Kugeloberfläche entfernt und liegen teilweise außerhalb des Messvolumens der NPM-Maschine. Die Kalibrierprozedur kann nicht vollständig ausgeführt werden, ohne dass die Endlagenschalter der NPM-Maschine ausgelöst werden. Durch das Auslösen der Endlagenschalter wird die Maschine gestoppt und die Kalibrierprozedur abgebrochen. Nach Rücksprache mit dem Hersteller konnten zwei Parameter identifiziert werden, welche die Entfernung der Parkpositionen verringern. Das Problem konnte aber insbesondere bei der Nutzung einer NMM-1 nicht vollständig gelöst werden. Die Durchführung der Tasterkalibrierung ist jedoch nicht zwingend erforderlich. Mit der Prozedur soll der effektive Tastkugelradius - die Differenz aus dem tatsächlichen Radius und der mittleren Schaftbiegung - bestimmt werden, welcher ausschließlich für die Tastkugelkompensation relevant ist. Der effektive Radius könnte auch im Rahmen der obligatorischen Sensorkalibrierung im Vorfeld der eigentlichen Inbetriebnahme mit I++DME ermittelt und manuell in der Tasterdatenbank der NPMM-Steuerung eingetragen werden.

Quindos7 ermöglicht die visuelle Darstellung von Messwerten von bis zu acht Dezimalstellen, wobei minimal 10 Pikometer aufgelöst werden können. Intern verarbeitet die Messsoftware die Daten jedoch mit höherer Auflösung und ermöglicht über seine flexible Programmierschnittstelle die vollaufgelöste Speicherung und Ausgabe von Messdaten für die externe Weiterverarbeitung, wie sie in Abschnitt 6.6 durchgeführt wurde. Leider weist Quindos7 bei der Festlegung der Fahrgeschwindigkeit eine Minimalbeschränkung aufgrund von Rundungseffekten auf. Werden Geschwindigkeiten von unter 1µm/s eingestellt, wird der eingegebene Wert aufgerundet und als 1µm/s angezeigt. Ob die Geschwindigkeit wie angegeben oder gerundet an die NPMM Steuerung übertragen wird, wurde nicht untersucht, da die Geschwindigkeit direkt im Server eingestellt werden kann und somit die Manipulation durch Quindos nicht zwingend erforderlich ist.

7.2.2 Calypso

Die Messsoftware Calypso DME (siehe Abbildung 79) - die I++DME-Version von Calypso von Zeiss Industrielle Messtechnik - stellt im Gegensatz zu Quindos7 keine eigene Kalibrierroutine für das Einmessen des Tasters bereit. Vielmehr wird davon ausgegangen, dass der Taster über den I++DME-Befehl ReQualify() mit einer serverseitig vorhandenen Funktion kalibriert wird bzw. die Kalibrierung vor der Übernahme durch Calypso erfolgt ist. Neben der Tastkugelkompensation können mit Calypso auch thermische Einflüsse kompensiert werden. Calypso liest dazu die momentane Temperatur von der NPMM-Steuerung aus und kompensiert die thermische Ausdehnung mit einem frei definierbaren Koeffizienten.

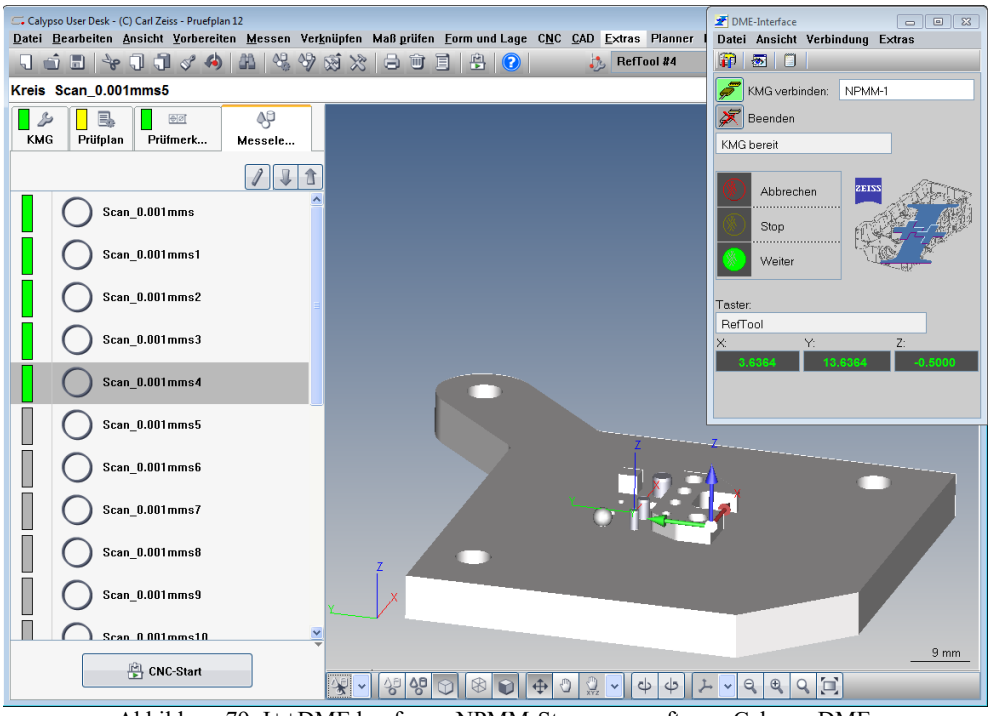


Abbildung 79: I++DME konforme NPMM-Steuerungssoftware Calypso DME

Der Arbeitsablauf zum Messen mit Calypso entspricht den sieben Schritten (siehe Abschnitt 7.1) und ist insofern mit dem Konzept von Quindos7 identisch. Calypso ist jedoch strenger organisiert und nutzt eine hierarchische Gliederung von Messplan, Formelementen und Einzelpunkt- bzw. Scanantastung. Dadurch wird eine hervorragende Übersichtlichkeit und intuitive Bedienbarkeit erreicht. Darüber hinaus besitzt Calypso im Vergleich zu Quindos eine fortschrittliche künstliche Intelligenz. So werden z. B. die Fahrwege durch die Definition eines Sicherheitsquaders um das Messobjekt automatisch generiert und die Ausrichtung eines neu erstellten (Bauteil-) Koordinatensystems automatisch aus der Ausrichtung der drei Grundelemente ermittelt.

Probleme mit der Verkürzung gemessener Werte existieren bei Calypso nicht. Über den Einstellungsdialog kann zentral die Anzahl der angezeigten Nachkommastellen für den Messplan festgelegt werden. Es sind maximal sechs Nachkommastellen einstellbar, was mit der I++DME-Basiseinheit Millimeter eine Auflösung von einem Nanometer ermöglicht. Selbstverständlich kann damit nicht das volle Potential von NPM-Maschinen ausgereizt werden. Demgegenüber ermöglicht Quindos7 in seiner Anwenderoberfläche die Darstellung von bis zu acht Dezimalstellen und die (Datei-)Ausgabe mit beliebiger Genauigkeit. Damit kann dieses I++DME-Messprogramm die Positionierauflösung von NPM-Maschinen hinreichend abdecken und ist Calypso im Hinblick auf die nanometrologisch korrekte Darstellung von Messdaten überlegen.

Der funktionelle Umfang von Calypso ist sehr hoch und umfasst viele Messprozeduren und standardisierte Auswertungsalgorithmen. Da die Berechnungen bei Calypso jedoch nicht im Sinne einer Programmierung über mathematische Operationen und Variablen erfolgt, sondern vielmehr über die vorgefertigten Basisfunktionen bereitgestellt wird, ist die Flexibilität bei weitem nicht so hoch wie bei Quindos. Die deutliche Ausrichtung der Messsoftware auf den industriellen Einsatz zum Zweck der Qualitätssicherung tritt dadurch stärker hervor.

Die große Stärke von Calypso ist die einfache und intuitive Bedienung, welche es auch ungeübten Nutzern ermöglicht, einen kollisionssicheren Messplan mit wenigen Mausklicks in kurzer Zeit zu erstellen. Von zentraler Bedeutung dafür ist jedoch das Vorhandensein einer Steuerkonsole am Positioniersystem. Calypso ermöglicht nämlich das Positionieren des KMGs mit Sollvorgaben nur

über einen versteckten Dialog. Während der Programmierung ist es damit nicht möglich, die vorgegebenen Sollpunkte anzufahren und auf eine korrekte Lage hin zu überprüfen. Eigens für die Inbetriebnahme mit Calypso wurde ein Softjoystick in die grafische Benutzeroberfläche der NPMM-Steuerung integriert, welcher es ermöglicht, die angeschlossene NPM-Maschine manuell zu positionieren und manuelle Punktmessungen über Buttons auszulösen und das Ergebnis an die I++DME-Messsoftware zu übertragen.

7.2.3 Metrolog

Nach Abschluss der Entwicklungsarbeiten für das Schnittstellensystem der NPMM-Steuerung wurden an die SIOS Messtechnik GmbH weiterführende Untersuchungen zur Anwendung mit Metrolog durchgeführt. Folglich ist der Vergleich von Funktionsumfang und Bedienkomfort mit den oben genannten KMG-Programmen nur eingeschränkt möglich.

Zusammengefasst bildet Metrolog in den Punkten Bedienung und Flexibilität in der Anwendung mit NPM-Maschinen einen guten Kompromiss zwischen den beiden oben genannten Messprogrammen. Die Nutzerschnittstelle besteht aus der zentralen CAD-Ansicht, über welche Formelemente und Messpunkte per Maussteuerung festgelegt werden können. Da Metrolog nicht über eine vergleichbare künstliche Intelligenz wie Calypso verfügt, müssen alle Fahrwege manuell programmiert werden. So ist der Nutzer schon bei der Erstellung des Messplans jederzeit darüber im Bild, wo und wie im Messvolumen navigiert wird. Dadurch wird der Messplan transparenter und fatale Kollisionen aufgrund der automatischen Berechnung aus einem fehlerhaften Sicherheitsbereich sind vermeidbar.

Metrolog ermöglicht die Überprüfung der Sollpunkteingaben durch Anfahren schon während der Erstellung des Messplans. Allerdings können gerade bei komplexen Messobjekten mit dieser reinen Maus- und Tastaturinteraktion schnell schwerwiegende Bedien- oder Eingabefehler auftreten und der Taster irreparabel beschädigt werden. Deshalb beherrscht auch Metrolog die Rückwärtssteuerung durch ein KMG-Bedienpult oder den Softjoystick.

Die Auflösung der Darstellung von Positionier- und Messdaten kann vom Nutzer beliebig angepasst werden. Es sind maximal acht Nachkommastellen möglich, womit Auflösungen im Bereich von bis zu zehn Pikometern erreicht werden. Die Gewinnung von Messdaten für die am CAD-Modell definierten Formelemente kann angepasst an die Messaufgabe flexibel mit verschiedenen Modi erfolgen. Die für die Qualitätssicherung relevanten Parameter werden mit vordefinierten Auswertefunktionen berechnet. Neben der Funktion zum Einmessen eines Tasters, ermöglicht das Programm auch die Übernahme eines kalibrierten Tasters durch die NPMM-Steuerung, welcher sofort für taktile Messungen angewendet werden kann.

7.2.4 Eignung von I++DME für nanometrologische Anwendungen

In Verbindung mit kommerziellen Steuerungs- und Messprogrammen stellt die NPMM-Steuerung eine große Bereicherung für die Nanomesstechnik dar. Die Anwendung von NPM-Maschinen kann durch die interaktive grafische Nutzerschnittstelle kommerzieller Software drastisch vereinfacht werden. Jedoch decken diese Koordinatenmessprogramme nur einen Teil des Anwendungsfeldes von NPM-Maschinen ab. Der Fokus liegt insbesondere auf der Messung von räumlich differenzierten Objekten. Eine Scanmessung mit Rastersonden ist zwar möglich, allerdings bieten die Programme keine Möglichkeiten, Messdaten als gleichabständige 2,5D-Repräsentation zu interpretieren. Gemessene Punkte werden stets einem Geometrieelement zugewiesen, aus welchem die KMG-Software ein globales Ausgleichselement ermittelt. Es existieren keine Funktionen zur Auswertung von Rastermessdaten wie z. B. die Extraktion eines Schnittprofils, die Ermittlung von Rauheitsparametern

oder die Bestimmung von Stufenhöhen bzw. Kantensteilheiten an einem Linienscan. Gerade diese Analysemethoden sind jedoch bei der Untersuchung von nanostrukturierten Oberflächen zwingend erforderlich.

Eine Möglichkeit 2,5D-Rasterdaten, welche mit einem I++DME Client gemessenen wurden, dennoch zu analysieren, ist die Datenexportfunktion der Messprogramme. Mit den Exportfunktionen der KMG-Software können die gemessenen Punkte als Datei gespeichert und mit einem Konvertierungsskript in gängige Formate für Rastersondenmessungen umgewandelt werden, welche in entsprechende Analyseprogramme importierbar sind. Diesbezüglich ist zu klären, ob unter Umständen I++DME-konforme Koordinatenmesssoftware mit Spezialisierung auf Freiformflächen (wie z. B. Holos von Renishaw) geeignete Funktionen für die Auswertung von 2,5D-Messdaten bereit stellt.

Ferner bilden die CAD-Modelle ein zentrales Element bei der Nutzung aller I++DME Messprogramme. Im Rahmen akademischer Forschung und Grundlagenuntersuchung an mikro- und nanostrukturierten Objekten und Oberflächen stehen derartige Modelle nur selten zur Verfügung. Oft soll die Formgebung eines Objekts überhaupt erst durch die Messung bestimmt werden. In diesem Fall besitzt die KMG-Software gegenüber der bisherigen Matlab-Ansteuerung wenige Vorteile. Nur mit einem CAD-Modell können die Fähigkeiten der Programme voll ausgeschöpft werden. Die Prüfplanung kann zwar auch ohne CAD-Modell erstellt werden, jedoch steigt dadurch insbesondere bei komplexen Messaufgaben die Wahrscheinlichkeit für eine Tasterkollision bei der Definition von Mess- und Freifahrpunkten. Ist kein CAD-Modell des Messobjekts vorhanden, sollte wegen der besseren Übersicht die Definition von Mess- und Freifahrpunkten mit dem Softjoystick der NPMM-Steuerung erfolgen. Dafür ist jedoch zwingend eine Möglichkeit zur Überwachung der Bewegung erforderlich, um Kollisionen zu vermeiden. Da das Positioniersystem wegen der besseren thermischen und akustischen Isolierung durch eine Haube abgedeckt wird, muss die visuelle Überwachung entweder durch den 3D-Übersichtssensor oder konventionelle Kameras gewehrleistet sein. So ist der Anwender jederzeit in der Lage, abzuschätzen, ob eine beabsichtigte Bewegung zu einer Kollision führt und kann die laufenden Bewegungen im Zweifel über den Dialog abbrechen.

Ein weiterer ungelöster Aspekt bei der Anwendung von KMG-Software ist, dass bei Mikrotastern die Form der Tastkugel nicht als ideal angenommen werden kann. Die Tastkugel muss durch geeignete Kalibrierverfahren vollständig charakterisiert werden und die Tastkugelkompensation muss anstatt eines festen Radius mit den Werten einer winkelabhängigen Radius-Lookup-Tabelle erfolgen. Insbesondere bei Messungen mit Rastersonden ist dieser Ansatz zwingend erforderlich, da die Form der Tastspitze durch ein geometrisches Grundelement nur unzulänglich beschrieben wird. Nicht zuletzt ist die Kalibrierung des Tastsystems bislang nicht hinreichend gelöst. Durch die Kalibrierung wird ein Bezug zwischen der am Taster gemessenen Größe und der Auslenkung des Tasters hergestellt. Da I++DME keine Möglichkeit vorsieht, dem Nutzer Rohdaten bereit zu stellen, kann die Kalibrierung nicht durch die I++DME Messsoftware erfolgen. In diesem Zusammenhang ist die Software Calypso sehr interessant. Sie existiert in zwei Varianten, mit proprietärer Schnittstelle für Zeiss-KMGs, welche die Kalibrierung von KMG-Sensoren ermöglicht, und in der I++DME Version. Diese Ausführung erfordert explizit das Ausführen der Kalibrierroutine durch die NPMM-Steuerung.

Die im Rahmen dieser Arbeit entworfene Steuerung für NPM-Maschinen bietet grundlegend die Möglichkeit der Unterstützung der internen Sensorkalibrierung. Da der maximal erlaubte Messbereich taktiler Sensoren jedoch nicht generell bekannt ist und keine allgemeingültigen Angaben zum Kalibrierobjekt getroffen werden können, wurde von der Implementierung der taktilen Sensor-Kalibrierung bislang abgesehen. Für die Experimente wurden die Taster vor den Messungen mit einem Matlab-Kalibrierskript kalibriert, die entsprechenden Parameter an die NPM-Maschine übertragen und das konfigurierte System anschließend an die NPMM-Steuerung übergeben. Die perspektivische

Portierung des Matlab-basierten Kalibrierverfahrens in C++ würde es ermöglichen, die Tasterkalibrierung in die NPMM-Steuerung zu integrieren und auf die Vorkonfiguration mittels Matlab zu verzichten.

Kapitel 8 Zusammenfassung

Im Rahmen der vorliegenden Arbeit konnte gezeigt werden, dass auf der Basis des holistischen Entwurfsansatzes eine skalierbare Softwareplattform für den Entwurf nanometrologischer Koordinatenmessgeräte gemäß der "Micro- und Nanometrologie Roadmap implementing Metrology in the European Research Area" (kurz iMERA) realisierbar ist. Der hohe Grad der automatisierten Datenerfassung bei der Untersuchung metrologischer Parameter setzt insbesondere bei NPM-Maschinen eine zentrale Steuerungskomponente voraus, welche in der Lage ist, alle Hardwaremodule der NPM-Maschine zu steuern und Objektmerkmale entsprechend der Aufgabenstellung zu erfassen. Diese Datenverarbeitungseinheit stellt damit den zentralen Kontenpunkt für die Verwaltung und Interaktion der einzelnen Teilsysteme dar, da sie dem Anwender die Funktionalität der metrologischen Hardware zugänglich macht.

Die vielfältigen Vorgaben der iMERA im Hinblick auf Multifunktionalität, multisensorielle Datenerfassung durch automatisierte Sensorwechseleinrichtungen und die Unterstützung mehrerer Rotationsachsen übersteigt jedoch die Möglichkeiten gegenwärtiger Strategien für den Entwurf eines Gesamtsystems maßgeblich. Im Rahmen der vorliegenden Promotionsschrift wurde ein Konzept erarbeitet, welche diese Herausforderungen durch einen neuartigen, modularen und frei skalierbaren Ansatz überwindet. Höchste Positionierauflösung in großräumigen Messvolumen gepaart mit der Forderung, unterschiedliche optische und taktile Sensoren automatisiert im Rahmen einer Messung zu verwenden, stellen die Entwickler vor noch ungelöste Aufgaben. Insbesondere der Entwurf und die Umsetzung der zentralen NPMM-Steuerung ist in diesem Hinblick eine unübersichtliche und nur schwer lösbare Aufgabe. Konventionelle Entwurfsansätze resultieren in einer monolithischen Software, welche ausschließlich für eine einzige Hardwarekonfiguration ausgelegt ist. Da in dieses Gefüge nur mit großem Aufwand funktionelle Änderungen oder metrologische Erweiterungen

eingebracht werden können, ist die nachträgliche hardwareseitige Veränderung eines bestehenden Systems nur mit hohem Aufwand realisierbar.

Das vorgestellte Entwurfskonzept für eine komponentenbasierte, frei skalierbare, heterogene und verteilt agierende NPMM-Zentralsteuerung beschreitet bereits bei der Strukturierung der Steuerungssoftware neue Wege. Statt des konventionellen monolithischen Aufbaus, bei welchem nur wenige a-priori definierte metrologische Teilsysteme angesprochen werden können, setzt dieses Konzept auf die Abstraktion der Teilsysteme durch Softwareschnittstellen. So wird es ermöglicht, die Steuerung von Subsystemen durch die zentrale Steuerungssoftware auf einer hohen Abstraktionsebene zu realisieren und die generischen Steuersequenzen erst durch hardwarenahe Schnittstellenadapter in von der Hardware interpretierbare Steuerungssequenzen umzusetzen. Für den Entwurf des Gesamtkonzepts wurde, soweit möglich, auf etablierte Schnittstellensysteme zurückgegriffen. Die anwenderseitige Anbindung erfolgt auf Basis einer Schnittstelle aus dem Umfeld der makroskopischen Koordinatenmesstechnik. Der Industriestandard *Inspection plus plus for Dimensional Measurement Equipment* (kurz I++DME) deckt alle metrologischen Anwendungsfälle für metrologische Analysen mit taktilen Sensoren ab und ermöglicht in Kombination mit der Erweiterung *Optical Sensor Interface Standard* (kurz OSIS) auch funktionell weitreichende Untersuchungen mit optischen Sensoren. Da auch diese Kombination nicht alle Anwendungsfälle der Nanometrologie abdeckt, wird die Anwenderschnittstelle um weitere proprietäre Funktionen ergänzt. In analytischen Abhandlungen wurden die messtechnische Eignung der vorgeschlagenen Schnittstellen mit Rücksicht auf die Anforderungen der Nanometrologie untersucht und der Eignungsnachweis für diese Anwendung erfolgreich erbracht. Weiterhin wurden die Datenverarbeitungsbasis digitaler Rechensysteme analysiert und der zu erwartende Fehlereinfluss durch die digitale Datenverarbeitung abgeschätzt.

Die Erprobung des zentralen Konzepts der Arbeit erfolgte im Rahmen der Umsetzung eines Softwaredemonstrators, welcher auf einer Nanopositionier- und Nanomessmaschine 1 (kurz NMM-1) basiert. Die skalierbare Demonstratorsteuerung verfügt über alle Softwarekomponenten einer iMERA-konformen NPM-Maschine. Damit erlaubt der modulare und skalierbare Aufbau, im Gegensatz zu einer konventionellen monolithischen Steuerung, funktionelle Teilsysteme mit entsprechenden Hardwaresystemen zu verbinden, ohne damit Einfluss auf die Funktionalität der übrigen Komponenten zu nehmen. Für die Erprobung aller Schnittstellen des Systems wurden unterschiedliche Sensorsysteme verwendet. Zum Teil konnte auf existente Sensoren zurückgegriffen werden, insbesondere bei den optischen Sensoren war es jedoch erforderlich, neue Systeme zu entwerfen und umzusetzen. Dazu wurde für jedes System zunächst eine Analyse des metrologischen Prinzips durchgeführt, um darauf aufbauend ein Konzept für ein intelligentes und autonom arbeitendes Sensorsystem zu entwickeln. Ebenso wie beim schnittstellenbasierten Entwurf der Zentralsteuerung wurde das Grundprinzip der Modularisierung auch bei der Konzeption der Sensorsysteme berücksichtigt, um die einfache Austauschbarkeit von Komponenten zu gewährleisten. Die gewonnenen Erkenntnisse und die modularen Entwurfskonzepte wurden in Softwarekomponenten überführt, durch welche alle metrologischen Aspekte von der Erfassung der Rohdaten, deren Verarbeitung zu Raumkoordinaten, bis hin zur Korrektur von systematischen Fehlern sowie deren Erfassung durch geeignete automatisierte Kalibrieralgorithmen abgedeckt werden.

Mit besonderem Hinblick auf das hohe Messdatenaufkommen bei nanometrologischen Analysen wurde das Laufzeitverhalten des Demonstrators eingehend untersucht. Aus den gewonnenen Daten kann gefolgert werden, dass die Datenverarbeitung auf konventioneller Datenverarbeitungshardware für den fortwährenden Messbetrieb mit einer NMM-1 bei maximaler Samplefrequenz hinreichend leistungsfähig ist. Abschließend erfolgte die Validierung des Gesamtsystems durch exemplarische Messungen mit taktilen und optischen Sensoren. Da die Datenverarbeitungskomponenten der

optischen Sensoren neu implementiert wurden, erfolgte im Rahmen der Systemvalidierung die eingehende Analyse dieser Systeme. Die Auswertung der Daten bestätigt die nanometrologische Eignung der optischen Sensoren und des modularen NPMM-Demonstrators. Durch die Verwendung des Schnittstellenstandards I++DME konnte mit dem Demonstrator ferner gezeigt werden, dass es möglich ist, mit I++DME-konformen Messprogrammen der makroskopischen Koordinatenmesstechnik nanometrologische Messungen auszuführen und die gewonnenen Daten mit diesen Programmen zu verarbeiten und darzustellen.

Das im Rahmen dieser Arbeit entwickelte, holistische Integrationskonzept für die Umsetzung einer modularen Steuerungseinheit für NPM-Maschinen stellt für die Nanopositionier- und Nanomesstechnik gleichzeitig im Hinblick auf mehrere Aspekte einen zukunftsweisenden Schritt dar. Durch die Abstraktion des Funktionsumfangs der hoch komplexen NPM-Maschinen auf die generische I++DME Schnittstelle ist es erstmals möglich, diese Systeme ohne Kenntnis über ihren Aufbau und ihre internen Funktionsabläufe nutzen zu können. Insbesondere sei an dieser Stelle auf die dadurch entstandene Möglichkeit zur Steuerung mit intuitiv bedienbarer Software aus dem Bereich der makroskopischen Koordinatenmesstechnik hingewiesen. Durch den Zugang zu den darin verankerten Bedienkonzepten eröffnen sich neue Möglichkeiten für die Nanomesstechnik. So ist es erstmals möglich, Messabläufe statt wie bisher prozedural nun modellbasiert und visuell an einem virtuellen Messobjekt zu planen und diesen Ablauf vor der eigentlichen Messung zu prüfen und gegebenenfalls zu optimieren. Dadurch können einerseits Kollisionen zwischen Messobjekt und Sensor nahezu vollständig ausgeschlossen werden und andererseits die Messzeit verringert werden. Darüber hinaus profitiert die Nanopositionier- und Nanomesstechnik von der weiten Durchdringung der makroskopischen KMT durch den Multisensoransatz. Jedes kommerziell erfolgreiche Steuerungsprogramm für makroskopische Anwendungen beherrscht die Planung und automatisierte Ausführung von Messungen mit mehreren Sensoren. Diese Möglichkeit wird dazu führen, dass mittelfristig neuartige Verfahren für die Kalibrierung, Kreuzreferenzierung und Datenfusion der im Vergleich zu KMT äußerst heterogenen sensorischen Ansätze geschaffen werden. All diese Aspekte wurden beim Design des Steuerungskonzeptes bereits berücksichtigt. Die konsequente Fortführung des Schnittstellengedanken ermöglicht es sehr einfach, die existierende Steuerungslösung zu einem späteren Zeitpunkt um derartige Funktionalität modular zu erweitern.

Letztendlich wird eine zunehmende Bedeutung von makroskopischen Freiformflächen für technische Anwendungen die schnelle und hochparallele Datenerfassung durch flächig messende optische Sensoren erfordern, wodurch die Fähigkeiten von in der Nanometrologie bereits gut etablierten optischen Verfahren in vollem Umfang ausgeschöpft werden können. Nicht zuletzt würde sich hieraus unter anderem auf Grund der einfachen Plan- und Steuerbarkeit von Messabläufen, gestützt auf die so erzeugbaren riesigen Datenvolumen mit bisher unbekannter lateraler Messdichte, ein weites Feld für völlig neue analytische Ansätze und Verfahren ergeben.

Anhang A: Erläuterungen zur Interpretation von UML-Diagrammen

Mit der wachsenden Komplexität nach prozeduralen Programmierparadigmen entwickelter Software, die z. B. mit ANSI C, Matlab, IDL und Pascal programmiert wurde, leidet mit zunehmender Codemenge die Übersicht und es wird für den Programmierer nahezu unmöglich den Überblick über das Zusammenwirken einzelner Module zu behalten.

Die *Objekt Orientierte Programmierung* ist ein Paradigma, welches es dem Programmierer ermöglicht, den Quellcode deutlich besser zu strukturieren bzw. zu kapseln, wodurch Teile der Software vollständig vor anderen Softwareteilen isoliert werden können. Dadurch ist es möglich, nur Funktionen und Variablen, welche unbedingt für die Interaktion mit anderen Programmteilen notwendig sind, für diese auch frei zu geben und so die Wahrscheinlichkeit für Fehlbedienungen und somit Programmierfehler drastisch zu reduzieren.

Die grundlegenden Kapselungselemente sind *Klassen*. Ein *Klasse* nutzt die Klassifizierbarkeit von physischen Objekten und kapselt Eigenschaften und Fähigkeiten einer eng umrissenen Gruppe von Gegenständen. Eine *Klasse* an sich ist nur eine abstrakte Beschreibung von Eigenschaften. Da eine Klasse nur eine Definition darstellt, kann diese an sich nicht direkt verwendet werden. Um eine *Klasse* verwenden zu können, müssen von ihr virtuelle *Objekte* instanziiert (erstellt) werden. Erst *Objekte* besitzen den in der *Klasse* festgelegten Funktionsumfang und können in einem Programm verwendet werden. In einem Programm können durchaus mehrere (Objekt-)Ausprägungen einer *Klasse* existieren.

Die Unified Modeling Language 2 (UML) stellt ein Hilfsmittel dar, welches, ähnlich einem Programmablaufplan für prozeduralen Quellcode, zu einer allgemein anerkannten Notationsform für die Dokumentation von Abhängigkeiten und Interaktionen in einem objektorientierten Programm avanciert ist. Die UML verfügt dazu über eine Vielzahl von Darstellungsformen, um den Softwareentwickler in allen denkbaren Designphasen bzw. der Dokumentation zu unterstützen.

Im Rahmen der Arbeit werden nur drei Notationsformen der UML verwendet, welche im Folgenden erläutert werden.

A.1 Klassendiagramm

Mit einem Klassendiagramm wird der Aufbau von objektorientierten Softwareentwürfen veranschaulicht. Eine Klasse oder deren Realisierung, das Objekte, repräsentiert oft ein physisch existierendes Objekt der realen Welt. Durch das Klassendiagramm wird der Aufbau von komplexen Objekten oder ganzen Softwaremodulen dargestellt. Die Erläuterung zum Lesen von Klassendiagrammen soll am Beispiel des vereinfachten Aufbaus eines virtuellen Personenkraftwagens erfolgen (vgl. Abbildung 80). Zunächst sollen jedoch die verwendeten Elemente erläutert werden.

Ein Kasten beschreibt eine Klasse. Die zusätzliche Bezeichnung <<*interface*>> gibt an, dass von dieser Klasse direkt keine Objekte erstellt werden dürfen. Ein Interface definiert nur eine Schnittstelle. Alle Linien in einem Klassendiagramm stellen Beziehungen dar. Linien mit einem Rhombus sind

Kompositionen (=Zusammensetzung). Die am Rhombus befindliche Klasse wird so als Teil oder Subsystem der anderen Klasse dieser Beziehung definiert. Die Zahlen neben der Komposition legen die Menge oder den Mengenbereich der beteiligten Objekte fest. Ein Stern steht für eine unendlich große Zahl.

Beziehungen mit einem Pfeil sind Spezialisierungen. Eine spezielle Klasse erweitert dabei eine allgemeinere Klasse um neue Eigenschaften oder Funktionen. Die Beziehung weist stets von der speziellen zur allgemeineren Klasse. Durch eine Spezialisierung wird der Mechanismus der Vererbung beschrieben.

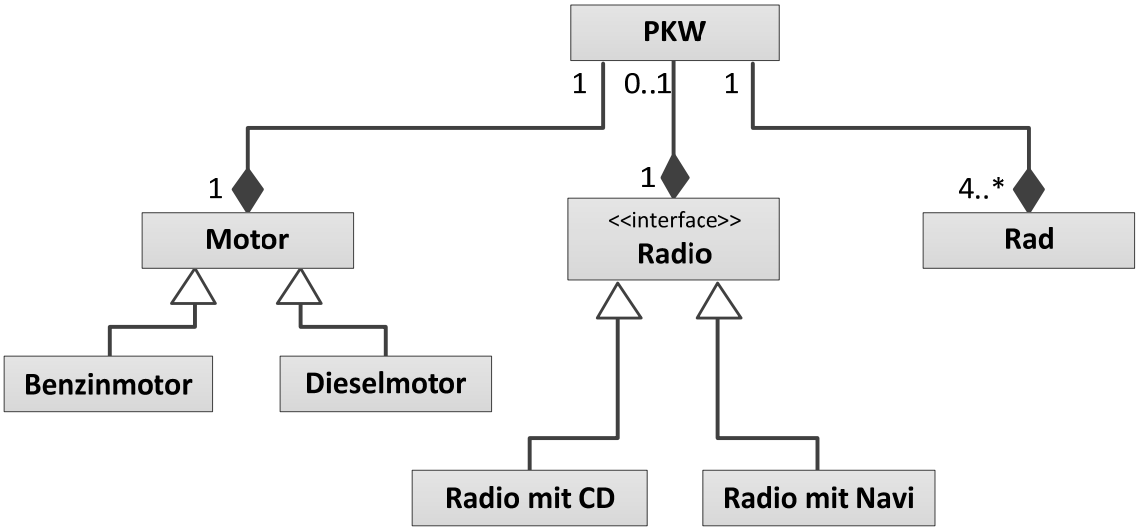


Abbildung 80: Klassendiagramm

Im Folgenden die Erläuterung zur obigen Grafik:

Ein PKW setzt sich in unserem vereinfachten Fall aus einem Teilsystemen *Motor* und mindestens vier Teilsystemen *Rad* zusammen. Als Optionen für den *Motor* existieren die Beiden Klassen *Benzin-* und *Dieselmotor*, welche aufgrund der Erweiterung der Klasse *Motor* jeweils auch ein Motor und somit mit dem *PKW* kompatibel sind. Weiterhin kann der *PKW* über eine *Schnittstelle* für ein *Radio* verfügen. In einem realen PKW würde diese Schnittstelle lediglich dem genormten Anschlussstecker für ein Radio entsprechen. Um die Funktionalität eines Radios überhaupt nutzen zu können werden Realisierungen der Schnittstellen, also ein *Radio mit CD* oder ein *Radio mit Navigationssystem* benötigt. Sie setzen die Schnittstelle zu einer instanziierbaren Klasse um, wovon wiederum Objekte erzeugt werden können.

A.2 Sequenzdiagramm

Sequenzdiagramme werden verwendet, um zeitliche Abläufe bzw. Interaktionen von Objekten darzustellen. Als Beispiel zur Veranschaulichung von Sequenzdiagrammen soll der Interaktionsablauf „an der Kasse mit EC-Karte bezahlen“ dienen. Siehe dazu die folgende Grafik und die Erläuterung darunter. Der Ablauf wird nicht im Detail beschrieben, da dieser Diagrammtyp recht einfach zu verstehen ist und es anzunehmen ist, dass der Leser mit dem Ablauf des dargestellten Vorgangs vertraut ist.

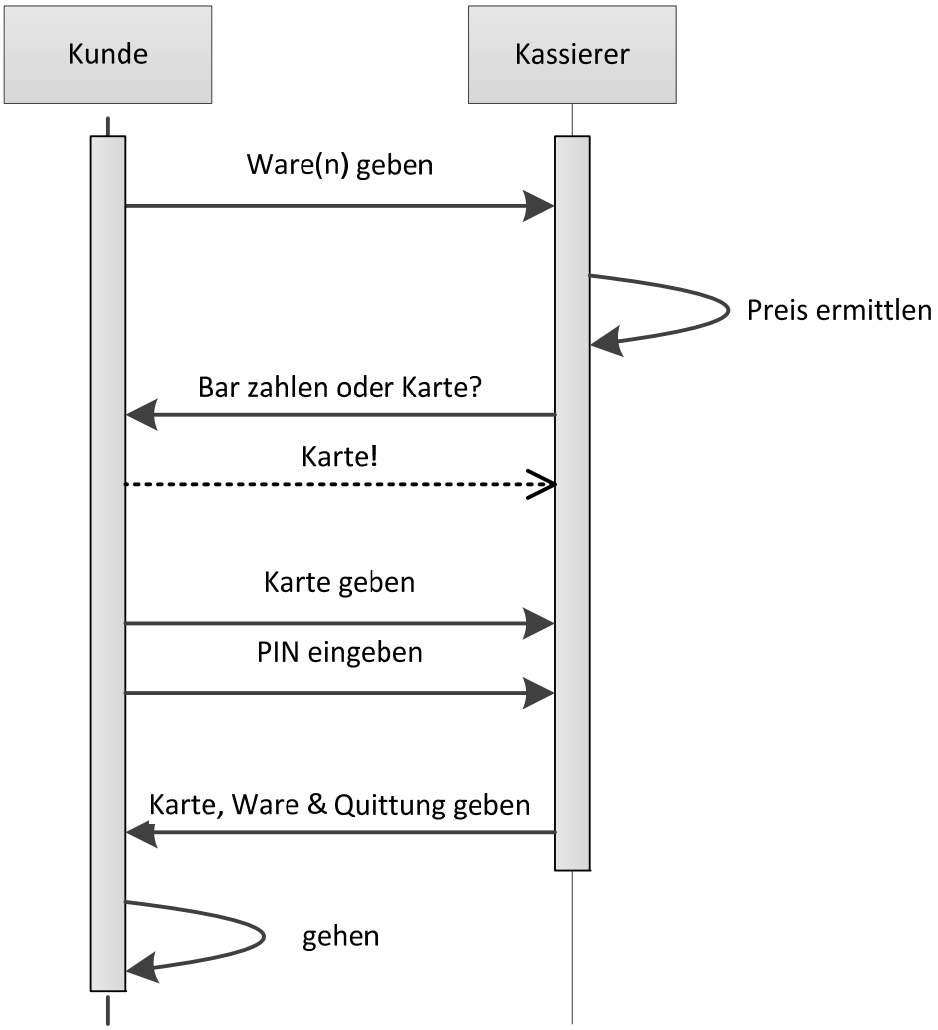


Abbildung 81: Sequenzdiagramm

Die beteiligten Objekte sind ein Kunde und ein Kassierer. Die vertikal von den Objekten ausgehenden gestichelten Linien werden als Lebenslinie bezeichnet. Sie stellen die Existenz des jeweiligen Objektes dar. Endet die Lebenslinie, wird das Objekt vernichtet. Durch die Lebenslinie wird weiterhin auch die zeitliche Abfolge der Interaktionen dargestellt. Je weiter eine Interaktion vom Ursprung der Lebenslinie, dem Objekt, entfernt liegt, desto später ist sie in der zeitlichen Abfolge einzuordnen. Für das obige Beispiel bedeutet dies, das Diagramm wird (zeitlich) von oben nach unten gelesen und die oberen Interaktionen erfolgen vor den darunter dargestellten.

Von der Lebenslinie ausgehende Pfeile stellen die Übermittelung von Nachrichten (allgemeiner Interaktionen) inklusive ihrer Übertragungsrichtung dar. Die Nachrichten können an fremde Objekte oder an das auslösende Objekt selbst gerichtet sein. Eine gestrichelt dargestellte Nachrichte ist die Antwort auf eine zuvor empfangenen Nachricht.

A.3 Verteilungsdiagramm

Verteilungsdiagramme werden verwendet um die strukturelle bzw. lokale Verteilung von Komponenten darzustellen. Mit dieser Spezialform der statischen UML-Diagramme wird die Verteilung von Softwarekomponenten über mehrere Rechnersysteme, bestehende Abhängigkeiten zwischen Komponenten und existente Kommunikationskanäle modelliert.

Das folgende Verteilungsdiagramm demonstriert ein Datenbanksystem mit der eigentlichen Datenbank auf einem Serversystem und einer entfernten Nutzeranwendung, welche auf die Datenbank zugreift.

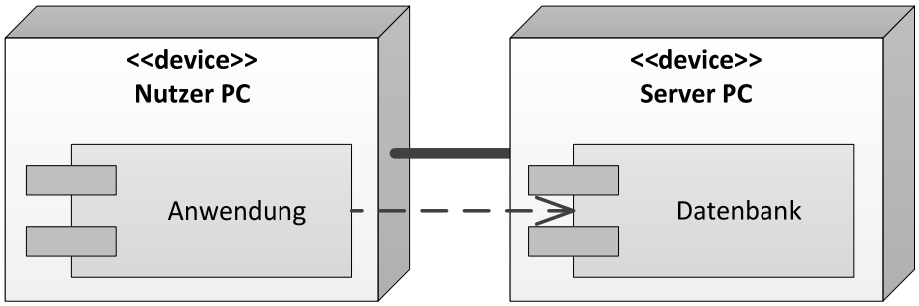


Abbildung 82: Verteilungsdiagramm

Die komplette Datenbankanwendung ist über zwei Rechnersysteme verteilt. Die eigentliche Datenbank befindet sich auf dem Server PC, der Programmteil mit Nutzerschnittstelle auf den Nutzer PC. Zwischen beiden Rechnern besteht eine Kommunikationsverbindung, welche durch eine geschlossene schwarze Linie modelliert wird. Die existentielle Abhängigkeit der Nutzeranwendung von der eigentlichen Datenbank wird durch den gestrichelten Pfeil dargestellt. Der Pfeil deutet dabei auf die Komponente, deren Existenz die notwendige Bedingung ist.

Anhang B: Anwenden des OSIS-Frameworks

Alle Grundfunktionen des OSIS-Frameworks sind vollständig implementiert und werden in Form einer auf *omniORB 4.1.4* basierenden Funktionsbibliothek bereit gestellt. Für die schlussendliche Implementierung einer OSIS-Schnittstelle muss der Anwender eine eigene Implementierung vom Framework ableiten und entsprechend der zu entwickelnden Applikation ergänzen. Das Framework wurde konzipiert, um den eigentlichen Entwicklungsprozess dahingehend zu vereinfachen, dass sich der Programmierer nicht mehr um die Kommunikation zwischen den OSIS-Objekten kümmern muss und sein ganzes Augenmerk auf die eigentliche Realisierung der sensoriellen Datenverarbeitung legen kann.

Die folgenden Abschnitte zum OSIS-Framework dienen der Dokumentation von anwendungsrelevantem Wissen über seinen Aufbau und die damit verbundene Funktionsweise. Es werden die notwendigen Details für die Nutzung des Frameworks vermittelt. Unter anderem wie Objekte angelegt und in das Framework integriert werden, die Art und Weise der Fehlerbehandlung und wie über die integrierten CORBA-Mechanismen auf entfernte Objekte zugegriffen werden kann. Die Dokumentation wird an geeigneten Stellen durch Quellcodefragmente veranschaulicht und ergänzt.

B.1 Erstellen von OSIS-Objekten und -Attributen

Alle inneren OSIS-Objekte werden mit dem C++Befehl `new` zur Laufzeit auf dem Heap angelegt, was den Austausch von Objekten zur Laufzeit und somit die Erweiterung und Anpassung des Frameworks ohne Eingriff in die kompilierte Bibliothek erlaubt. Diese Vorgehensweise erlaubt es, mit nur einer Bibliothek ein *OSISInterface* Objekt mit mehreren unterschiedlichen sensorspezifischen Implementierungen von *Tool* zu verwenden.

Das Einfügen von neuen Attributen (abgeleitet von *PropertyDescription*) oder Objekten (abgeleitet von *CommonServices*) ist im gesamten Framework durch Überladung der vorhandenen Klassen möglich. Die neu eingefügten Objekte und Attribute müssen eine eindeutige und von den bereits vorhandenen Eigenschaften bzw. Objekten verschiedene Bezeichnung besitzen. Im Konstruktor des überladenen Objektes sind die neuen Objekte mit der Methode

```
void cComnServ::signInObj(cComnServ *pNewObj);
```

und neue Eigenschaften mit

```
void cComnServ::signInProp(cPropDescr &newProp);
```

zu registrieren. Durch die Registrierung können die neuen Elemente im OSIS-Framework gefunden werden und sind über die CORBA-Mechanismen auch für andere Programmteile zugänglich.

Der Grund dafür, dass Objekte als Zeiger und nicht wie Attribute als Referenz eingetragen werden, ist die einfachere Verwendung der Registrier-Methoden. Wie oben beschrieben, sollten wegen der höheren Laufzeitflexibilität alle Objekte auf dem Heap angelegt werden. Diese Objekte werden über

einen Zeiger adressiert. Da Eigenschaften generell an die Objekte gebunden sind, besteht keine Notwendigkeit, sie im einmal kompilierten Programm zu verändern, weshalb sie immer auf dem Stack angelegt werden können.

B.2 Fehlerbehandlung im Framework

Die von OSIS definierte Fehlerklasse kann im gesamten Framework von jeder OSIS-Klasseninstanz geworfen werden. Zu diesem Zweck sind die OSIS-Klassen *CommonService* und *PropertyDescription* von der gemeinsamen Basisklasse *ErrorMessenger* abgeleitet.

Der *ErrorMessenger* besitzt nur eine Methode und zwei statische Attribute, einen Semaphor zum Anzeigen von Ereignissen und eine Liste zum Speichern der aufgetretenen Fehler. Die statischen Attribute sind im gesamten Framework nur einmal vorhanden, d. h. alle OSIS-Klasseninstanzen teilen sich eine Fehlerliste und einen Semaphor. Soll an einer beliebigen Stelle der OSIS-Implementierung ein Fehler geworfen werden, erfolgt dies durch einen Aufruf der Methode

```
void cErrorMessenger::throwError(iError_impl &err);
```

mit dem Fehlerobjekt als Parameter. Die Methode fügt den übergebenen Fehler am Ende der gemeinsamen Liste ein und signalisiert über den Semaphor, dass ein Fehler aufgetreten ist.

Die Grundlage für die Behandlung der Fehler bildet eine weitere Klasse namens *ErrorHandler*. Der *ErrorHandler* startet im Konstruktor automatisch einen Thread, welcher bis zur Signalisierung des statischen Semaphors ruht. Wird der Semaphor ausgelöst, tritt der Thread in die Fehlerbehandlungsmethode ein. Da keine generelle Antwort auf die Art der Fehlerbehandlung in einer späteren OSIS-Implementierung gegeben werden kann, wurde die Fehlerbehandlungsmethode als abstrakte Methode definiert. So wird der Programmierer gezwungen, eine Fehlerbehandlung zu programmieren. Anderenfalls kann das Programm nicht kompiliert werden.

Obwohl die Fehlerbehandlung im OSIS-Framework zentral geschehen soll, erben zwei Objekte (*CMMApp* und *OSISInterface*) die abstrakte Methode vom *ErrorHandler*. Die Einführung von zwei *ErrorHandler*-Instanzen ist erforderlich, da die OSIS-Objekte (*CMMApp* und *OSISInterface*) auf zwei autarke Programmteile verteilt sind. Die Fehlerbehandlungsmethode von *OSISInterface* dient ausschließlich dazu, die asynchron auftretenden Fehler zu sammeln und an *CMMApp* weiter zu leiten. Die Signalisierung erfolgt über das Setzen des Attributs *IsError* und dem Aufrufen der *CMMApp*-Schnittstellenmethode *HandleEvent()* mit einem *StatusChangedE*-Objekt mit *IsError* als Parameter.

Da die Fehlerbehandlungsmethode von *OSISInterface* vollständig implementiert ist, darf sie vom Anwender nicht überladen werden.

B.3 Zugriff auf verteilte Objekte über das CORBA-Framework

Um das OSIS-Framework in eine eigene Messapplikation zu integrieren, müssen Verbindungen und Schnittstellen zu eigenen Datenverarbeitungsalgorithmen und der verwendeten Sensorhardware geschaffen werden. Dazu sind gegebenenfalls neue Attribute und Objekte in das Framework zu integrieren. Die hauptsächliche Arbeit besteht jedoch in der Erstellung einer an Sensorhardware und Datenverarbeitung angepassten Überladung der Klasse *Tool* und einer spezialisierten Klasse *CMMApp*, welche das Mess- und Positionierregime repräsentiert. Im Folgenden wird die allgemeine Nutzung von verteilten Objekten demonstriert.

Über die Rechner- oder Programmgrenze hinweg können nur Referenzen, jedoch nicht die Objekte selbst ausgetauscht werden. Auf die entfernten Objekte kann jedoch über die Referenz vollständig zugegriffen werden. Die Art eines Zugriffs ist mit einem Objektzugriff über einen Zeiger vergleichbar. Da Referenzen jedoch keine echten Zeiger sind, entfallen die symbolischen Zeigeroperationen =, ==, != für Zuweisung und Vergleich. Diese Operationen werden durch spezielle CORBA-Methoden zur Verfügung gestellt. Weitere Erläuterungen dazu können der *omniORB* Dokumentation [93] entnommen werden.

Das folgende Codefragment veranschaulicht die Abfrage eines Attributes von einem entfernten Objekt. Der Quellcode entstammt einer möglichen *CMMApp*-Implementierung.

```
replyWord notValid;
tBoolean IsError;
// request the current error state
notValid = rOSISif->GetBoolProp("IsError", IsError);
if(notValid == 0 && IsError == OK)
   rOSISif->StartScanMode();
```

Über die Objektreferenz von rOSISif des *OSISInterfaces* wird auf die *CommonServices*-Methode *GetBoolProp()* zugegriffen, der Zustand des Fehlerflags abgefragt und in Abhängigkeit vom Resultat mit einer Messung begonnen.

Natürlich können neben den einfachen Datentypen wie Strings, Boolean usw. auch Referenzen von anderen Objekten ermittelt werden. Im folgenden Codefragment wird ausgehend von der *OSISInterface*-Referenz über eine Folge von Referenzabfragen die momentan eingestellte Leistung des Lasers ermittelt:

```
iCommonServices_var rActTool, rLaser;
replyWord notValid;
tInteger power;

// gain the active tool refernce:
notValid = rOSISif->GetActiveTool(rActTool);
if(notValid != 0)
   throwError(cOSISError_5312("ActiveTool not found",
                              __FUNCTION__          ))
// access the laser refernce:
notValid = rActTool->findObject("Laser", rLaser);
if(notValid != 0)
   throwError(cOSISError_5312("Laser not found",
                              __FUNCTION__     ))
// request the current laser power output:
notValid = rLaser->GetIntProp("Power", power);
if(notValid != 0)
   throwError(cOSISError_5312("Power not found",
                              __FUNCTION__     ))
```

An diesem Beispiel wird neben dem Werfen von Fehlerobjekten verdeutlicht, wie mit einer initialen Objektreferenz über vorgegebene OSIS-Methoden auf subsequente Objekte und Attribute zugegriffen werden kann. Wie eine initiale Referenz vom *omniNameService* ermittelt werden kann, wird im folgenden Codefragment verdeutlicht.

```
CORBA::ORB_var orb;
CosNaming::NamingContext_var nc;
CORBA::Object_var obj1, obj2;
OSIS::iCMMApp_var rCMMApp;
char *szNS = "NameService";

orb = CORBA::ORB_init(0, NULL);
obj1 = orb-> resolve_initial_references(szNS);
nc = CosNaming::NamingContext::_narrow( obj1.in() );
CosNaming::Name name;
name.length(1);
name[0].id = CORBA::string_dup("CMMApp");
obj2 = nc->resolve(name);
rCMMApp = OSIS::iCMMApp::_narrow( obj2.in() );

// make a single remote request:
replyWord notValid;
tBoolean isBusy;
notValid = rCMMApp->GetBoolProp("IsBusy", isBusy);
```

Bei der Instanziierung von *CMMApp*- und *OSISInterface*-Derivaten meldet das Framework die Objekte automatisch als „CMMApp“ bzw. „OSISInterface“ beim *omniNameService* an und entfernt die Einträge beim Abbau der Objekte wieder aus dem IOR-Register. Somit sind die beiden Initialreferenzen stets gültig und für den Anwender einfach zu ermitteln.

Literaturverzeichnis

[1] Semiconductor Industry Association, "The International Technology Roadmap for Semiconductors," 2011.

[2] J Geelan. (2008, May) Ulitzer. [Online]. http://jeremygeelan.sys-con.com/node/557154

[3] M Fuechsle et al., "A single-atom transistor," *Nature Nanotechnology*, vol. 7, pp. 242–6, Feb. 2012.

[4] R. Kleindienst, R. Kampmann, and S. Sinzinger S. Stoebenau, "Hybrid optical (freeform) components-functionalization of nonplanar optical surfaces by direct picosecond laser ablation," *Applied Optics*, vol. 50, no. 19, pp. 3221-3228, 2011.

[5] C. Liu, *Foundations of MEMS*, 2nd ed. Upper Saddle River, USA: Pearson Education Limited, 2011.

[6] R. Mastylo and E. Manske, Oberflächenscann an einer mikrostrukturierten Linse des IOT Stuttgart, 2011, interne Quelle SFB622.

[7] Informationsdienst Wissenschaft. (2009, Apr.) idw-online.de. [Online]. http://idw-online.de/pages/de/news311852

[8] iMERA Metrology Roadmaps. (2007, May) iMERA Metrology Roadmaps. [Online]. http://www.technology-roadmaps.eu/doku.php?id=micro_and_nano

[9] E Manske, G Jäger, T Hausotte, and R Füßl, "Recent developments and challenges of nanopositioning and nanomeasuring technology," *Meas. Sci. Technol.*, vol. 23, no. 7, p. 074001, Juli 2012.

[10] EURAMET e.V. (2012, Oct.) Science and Technology Roadmaps for Metrology. [Online]. http://www.euramet.org/fileadmin/docs/Publications/roadmaps/EURAMET_Science_and_Technology_Roadmaps_for_Metrology.pdf

[11] R Feynman, "There is plenty of room at the bottom," *Engineering and Science*, 1960.

[12] K E Drexler, *Nanosystems: Molecular Machinery, Manufacturing and Computation*.: Wiley, 1992.

[13] E. Abbe, *Beiträge zur Theorie des Mikroskops und der mikroskopischen Wahrnehmung*. Frankfurt: Univ.-Bibliothek Frankfurt am Main, 1873.

[14] E. Hecht, *Optik*. München: Oldenburg, 2009.

[15] R. Mastylo, E. Manske, and G. Jäger, "Development of a focus sensor and its integration into the nanopositioning and nanomeasuring machine," in *Proc. of OPTO 2004*, Nürnberg, 25.-27. Mai 2004.

[16] U. Neuschaefer-Rube, K. Wendt, and W. Ehring, "Optische Sensoren für die Koordinatenmesstechnik - Prinzipien und Prüfung," *PTB-Mitteilung*, vol. 117, no. 4, pp. 380-389, 2007.

[17] C.P., Züst, R. Keferstein, M. Marxer, and O. Gächter, "Optische Sensoren auf Koordinatenmessgeräten," in *VDI-Berichte*, Erlangen, 9.-10. Oktober 2001, pp. 181-192.

[18] T. Machleidt, D. Kapusi, K.-H. Franke, and E. Manske, "Depth from focus (DFF) utilizing the large measuring volume of a nanopositioning and nanomeasuring machine," , Berlin, 18.-19. Mai 2010.

[19] D Kapusi, T Machleidt, K-H Franke, and R Jahn, "White light interferometry in combination with a nanopositioning nad nanomeasuring machine (NPMM)," *Proc. of SPIE - Optical Measurement Systems for Industrial Inspection*, vol. 6616, no. 5, p. 07, Juni 2007.

[20] "New reflected-light microscope for viewing unstained brain and ganglion cells," *Science*, vol. 157, no. 3786, pp. 305-7, 1967.

[21] R Mastylo, *Optische und taktile Nanosensoren auf der Grundlage des Fokusverfahrens für die Anwendung in Nanopositionier- und Nanomessmaschinen*, Dissertation, Ed. Ilmenau: TU-Ilmenau, 2012.

[22] G Binnig, C F Quate, and Ch Gerber, "Atomic Force Microscope," *Phys. Rev. Ltr.*, vol. 9, no. 56, pp. 930-3, Mar. 1986.

[23] P Barth, *Hochauflösende Strukturierung von Siliziumoberflächen mittels Mikrokontaktdruck-Technik untersucht mit Rasterkraftmikroskopie*, Dissertation ed. Ulm: Universität Ulm, 2006.

[24] F Meli and R Thalmann, "Das Metrologie-Rasterkraftmikroskop," *OFMET Info*, vol. 1, no. 6, pp. 1-7, 1999.

[25] R.J. Hocken, D.L. Trumper, and C. Wang, "Dynamics and Control of the UNCC/MIT Sub-Atomic Measuring Machine," *CIRP Annals - Manufacturing Technology*, vol. 50, no. 1, pp. 373-376, 2001.

[26] J. Kramar et al., "The Molecular Measuring Machine," , Hsinchu, Taiwan, 30. Dezember 1998.

[27] R. Leach et al., "Advances in traceable nanometrology at the National Physical Laboratory," *Nanotechnology*, no. 12, 2001.

[28] A. Weckenmann and B. Gawande, *Koordinatenmesstechnik: Flexible Meßstrategien für Maß, Form und Lage*. München: Carl Hanser Verlag München, 1999.

[29] K. Wendt, C. Keck, and H. Schwenke, "Prüfprozesseignung von Inline-Messtechnik im Karosseriebau," *PTB-Mitteilungen*, vol. 117, no. 4, pp. 417-424, 2007.

[30] T. Pfeifer, *Fertigungsmesstechnik*. München: Oldenburg Verlag München, 2001.

[31] C.P. Keferstein and W. Dutschke, *Fertigungsmesstechnik: Praxisorienterite Grundlagen, moderne Messverfahren*. Wiesbaden: B.G. Teubner Verlag / GWV Fachverlag GmbH, 2008.

[32] H.-J. Gefatter and U.: Grünhaupt, *Handbuch der Mess- und automatisierungstechnik in der Produktion*. Heidelberg: Springer-Verlag, 2006.

[33] R Keller, "I++DME: Ziele, Konzepte und deren Umsetzung in Produkten," *VDI-Z*, vol. 148, no. 10, p. 2, Oktober 2006.

[34] Wenzel Metromec AG. (2011, Mai) Metrosoft CM - I++DME. [Online]. http://www.metromec.ch/images/stories/pdf/I++_CM390_d.pdf

[35] Renishaw. (2009, Juli) I++DME specification version 1.7. [Online]. http://resources.renishaw.com/details/I%20%20DME%20specification%20version%201.7(19564)

[36] Ian Gorton, *Essential Software Architecture*. Berlin: Springer Verlag, 2006.

[37] T. Rijul, *Ultra Precision Coordinate Measuring Machine ; Design, Calibration and Error Compensation*. Delft University of Technology, 2001.

[38] R. Donker, I. Widdershoven, and H. Henny Spaan, "Realization of Isara 400: a large measurement volume ultra-precision CMM," in *Asian Symposium for Precision Engineering and Nanotechnology*, 2009.

[39] K.-C. Fan, Y. Fei, X. Yu, W. Wanga, and Y. Chen, "Study of a noncontact type micro-CMM with arch-bridge and nanopositioning stages," *Robotics and Computer-Integrated Manufacturing*, no. 23, pp. 276–284, 2006.

[40] U. Brand, T. Kleine-Besten, and H. Schwenke, "Development of a special CMM for dimensional metrology on microsystem components," , Scottsdale, Arizona, 22.-27. Oktober 2000, pp. 542–546.

[41] Carl Zeiss AG Industrielle Messtechnik. (2010, Dezember) F 25 Auf Nanometer genau. [Online]. http://www.zeiss.de/f25

[42] A.J.M. Moers, van Riel M.C.J.M., and E.J.C Bos, "Design and Verification of the TriNano Ultra Precision CMM," in *Proc. of ISC 2011*, Ilmenau, 2011.

[43] G. N. Peggs, A. J. Lewis, and S. Oldfield, "Design for a compact high-accuracy CMM," in *CIRP Annals*, vol. 48, 1999, pp. 417-420.

[44] G. Jäger, E. Manske, T. Hausotte, and H.J. Büchner, "Nanomessmaschine zur abbefehlerfreien Koordinatenmessung," *tm - Technisches Messen*, vol. 67, no. 7-8, 2000.

[45] E. Abbe, "Messapperate für Physiker," *Zeitschrift für Instrumentenkunde*, no. 10, pp. 446-7,

1890.

[46] G. Jäger et al., "Nanomeasuring Technology - Nanomeasuring Machine," , Washington D.C., Nov. 2001.

[47] G. Jäger et al., "Traceable Nanometrology Realized by Means of Nanopositioning and Nanomeasurements Machine," *Key Engineering Materials*, no. 381-382, 2008.

[48] G. Jäger, E. Manske, T. Hausotte, and H.-J. Büchner, "The Metrological Basis and Operation of Nanopositioning and Nanomeasuring Machine NMM-1," *tm - Technisches Messen*, vol. 76, no. 5, 2009.

[49] T Hausotte, *Nanopositionier- und Nanomessmaschinen*. Ilmenau: TU-Ilmenau, 2002.

[50] SIOS Messstechnik GmbH, Datenblatt Nanopositionier- und Messmaschine NMM-1, 2007.

[51] B. Percle, J. Klöckner, E. Manske, and W. Fengler, "Signal and Data Processing in High-Precision Measuring Machines - A Case Study of Next-Generation NPMMs," in *Proc. of ISC 2011*, Ilmenau, 2011.

[52] A. Tibrewala, N. Hofmann, A. Phataralaoha, G. Jäger, and S. Büttgenbach, "Development of 3D Force Sensors for Nanopositioning and Nanomeasuring Machine," *Sensors*, no. 9, 2009.

[53] F. G. Balzer, T. Hausotte, N. Dorozhovets, E. Manske, and G. Jäger, "Tactile 3D microprobe system with exchangeable styli," in *Meas. Sci. Technol.*, vol. 22, 2011, p. 09318.

[54] XPress Precission Engineering B.V. (2011) Gannen XP. [Online]. http://www.xpresspe.com/probe_xm1.htm

[55] S. Bütefisch, S. Büttgenbach, T. Kleine-Besten, and U. Brand, "Micromechanical three-axial tactile force sensor for micromechanical characterization," *Microsystem Technologies*, vol. 4, no. 7, pp. 171-174, 2001.

[56] G. N. Peggs, A. J. Lewis, and S. Oldfield, "Design for a compact high-accuracy CMM," in *CIRP Annals*, vol. 48, 1999, pp. 417-420.

[57] S. Bütefisch, G. Dai, H.-U. Danzebrink, and U. Büttgenbach, S. Brand, "Ultra high Precision 3D Microprobe for CMM Applications," in *Mikrosystemtechnik Kongress 2009*, Berlin, 2009, p. 1.16.

[58] H. Hertz, "Über die Berührung fester elastischer Körper," *Journal für die reine und angewandte Mathematik*, no. 92, pp. 156-71, 1881.

[59] E. Manske et al., "New applications of the nanopositioning and nanomeasuring machine by using advanced tactile and non-tactile probes," *Meas. Sci. Technol*, no. 520-527, 2007.

[60] N. Dorozhovets, T. Hausotte, N. Hofmann, E. Manske, and G. Jäger, "Development of the interferometrical scanning probe microscope," , SPIE: Optics & Photonics; San Diago, Calif.,

16.-17. August 2006.

[61] A. Weckenmann, G. Peggs, and J. Hoffmann, "Probing systems for dimensional micro- and nano-metrology," *Meas. Sci. Technol.*, no. 17, pp. 504-509, 2006.

[62] D. Kapusi, T. Machleidt, E. Manske, K.-H. Franke, and R. Jahn, "Whilte light interferometry utilizing the large measuring volume of a nanopositioning and nanomeasuring machine," in *Proc. of International Colloquium on Surfaces, 12 (Chemnitz)*, Aachen, 28.-29. Januar 2008.

[63] I. Schmidt, T. Hausotte, U. Gerhardt, E. Manske, and G. Jäger, "Investigations and calculations into dectreasing the uncertainty of a nanopositioning and nanomeasuring machine (NPM-Machine)," *Mea. Sci. Technol.*, vol. 18, no. 2, 2007.

[64] A Weckenmann, A Schuler, Ngassam, and RJB, "Enhanced measurement of steep surfaces by slope-adapted sensor tilting," *Meas. Sci. Technol.*, vol. 23, no. 7, p. 0774007, Juli 2012.

[65] E Manske, in *6th Sino-German Symposium on Micro- and Nano Production, Measurement an Application*, Braunschweig, 09.2012.

[66] A. Oeder et al., "Optische Manipulation von Mikropartikeln in mikro-opto-fluidischen Systemen," in *DGaO-Proceedings*, 2008, p. B27.

[67] A. Oeder, C. Bauer, S. Stoebenau, and S. Sinzinger, "Optimierte Strahlformungs- und Fokussieroptik zur optischen Mikromanipulation," in *DGaO-Proceedings*, 2010, p. A37.

[68] Y. Chen and A. Pepin, "Nanofabrication: Conventional and nonconventional methods," in *ELECTROPHORESIS*, vol. 22, 2001, pp. 187-207.

[69] R. D. Piner, J. Z. F. Xu, S. Hong, and C. A. Mirkin, ""Dip-Pen" Nanolithography," *Science Magazine*, vol. 283, no. 5402, pp. 661-663, 1999.

[70] Nano-Standard. (2009, Feb.) Roadmaps for the Standardization and Metrology of Nanotechnologies. [Online]. http://www.ebn.din.de/sixcms_upload/media/2929/NANOSTRAND.pdf

[71] G. Dai, "Auf dem Weg zur 3D-Nanometrologie," *PTB news 09.1*, no. 1, p. 1, April 2009.

[72] Institute of Electrical and Electronical Engineering, *IEEE Standard for Binary Floating-Point Arithmetic*. New York, USA, 1985.

[73] Microsoft Corporation. (2010, Dezember) Run-Time Dynamic Linking. [Online]. http://msdn.microsoft.com/en-us/library/ms685090.aspx

[74] Microsoft Corporation. (2011, April) Critical Section Objects. [Online]. http://msdn.microsoft.com/en-us/library/ms682530(VS.85).aspx

[75] O. Bouky, "OSIS - Optical Sensor Interface Standard," *Innovation SPEZIAL Messtechnik*, no. 5, 2003.

[76] C.P. Keferstein and R. Züst, "Minimizing technical anf financial risk when integrating optical sensors for in process measurement," Institute of Quality Technology, Interstate University of Applies Sciences of Technology Buchs, NTB, Buchs, Switzerland, 2004.

[77] *Optical Sensor Interface Standard - Workgroup 2: Data integration*., 2003.

[78] Qtechnology A/S. (2011, Januar) Camera platforms. [Online]. http://www.qtec.com/en/productssolutions/cameras/platforms

[79] MaxxVision GmbH. (2011, Mai) Prospekt Sony Smart Kamera. [Online]. http://www.maxxvision.com/fileadmin/content/Produkte/Kameras/Smart_Kameras/Downloads/Sony_Smart_Kamera_Prospekt.pdf

[80] Microsoft Corporation. (2011, Januar) Windows Embedded. [Online]. http://www.microsoft.com/windowsembedded/de-de/default.mspx

[81] J. D. Dionne and M. Durrant. (2010, Juli) µClinux - Embedded Linux/Microcontroller Project. [Online]. http://uclinux.org/

[82] D. Harms, *Anwendung des Betriebssystems µClinux für bildverarbeitende Sensoren im industriellen Einsatz*. Ilmenau: TU Ilmenau, 2009.

[83] Microsoft Corporation. (2010, Dezember) Windows Sockets 2. [Online]. http://msdn.microsoft.com/en-us/library/ms740673.aspx

[84] J. E. White, *A High-Level Framework for Network-Based Resource Sharing*.: National Computer Conference, 1976.

[85] O. Haase, *Kommunikation in verteilten Anwendungen - Einführung in Sockets, Java RMI, Corba und Jini*. München: R. Oldenburg, 2001.

[86] (2014, June) Java Remote Method Invocation - Distributed Computing for Java. [Online]. http://www.oracle.com/technetwork/java/javase/tech/index-jsp-138781.html

[87] Sun Microsystems. (2003, Januar) RMI Specification. [Online]. http://java.sun.com/j2se/1.4.2/docs/guide/rmi/spec/rmiTOC.html

[88] Object Management Group. (2008, Januar) Common Object Request Broker Architecture (CORBA) Specification, Version 3.1. [Online]. http://www.omg.org/spec/CORBA/3.1/Interfaces/PDF/

[89] Inc. Object Management Group. (2007, Feb.) OMG IDL Syntax and Semantics. [Online]. http://www.omg.org/cgi-bin/doc?formal/02-06-07.pdf

[90] A. Punder and K. Römer, *Mico - An Open Source CORBA Implementation*. San Diego: Academic Press, 2000.

[91] Distributed Object Computing Group. (2010, August) Real-time CORBA with TAO (The ACE

ORB). [Online]. http://www1.cse.wustl.edu/~schmidt/TAO.html

[92] A. Punder, K. Römer, and K. Geihs, *MICO: An Open Source CORBA Implementation*. Heidelberg: Dpunkt Verlag, 2000.

[93] D. Grisby. (2010, Dezember) omniORB : Free CORBA ORB. [Online]. http://omniorb.sourceforge.net/

[94] Microsoft Corporation. (2014, Jan.) [MS-DCOM]: Distributed Component Object Model (DCOM) Remote Protocol Specification. [Online]. http://download.microsoft.com/download/a/e/6/ae6e4142-aa58-45c6-8dcf-a657e5900cd3/[MS-DCOM].pdf

[95] The Open Group. (2012, May) DCE Home Page. [Online]. http://www.opengroup.org/dce/

[96] (2014, Januar) Einführung in COM. [Online]. http://msdn.microsoft.com/de-de/library/cc451377(v=vs.71).aspx

[97] Microsoft Corporation. (2014, Jan.) History of DCOM. [Online]. http://msdn.microsoft.com/de-de/library/6zzy7zky.aspx

[98] U Lindquist, "Noch Hürden auf dem Weg zur perfekten Middleware," *Computerwoche*, no. 37, 1997.

[99] Open Group. (2009, Dezember) COMsecure. [Online]. http://www.opengroup.org/comsource

[100] P.E. Chung et al., "DCOM and CORBA Side by Side, Step by Step and Layer by Layer," *C++ Report*, no. September, 1997.

[101] M. Horstmann and M. Kirtland. (1997, Juli) Microsoft Developer Network. [Online]. http://msdn.microsoft.com/de-de/library/ms809311(en-us).aspx

[102] Microsoft Corporation. (2014, Jan.).NET Remoting. [Online]. http://msdn.microsoft.com/de-de/library/bb979191.aspx

[103] F. P. León and J. Beyerer, "Oberflächencharakterisierung durch morphologische Filterung," *tm - Technisches Messen*, no. 12, 2005.

[104] G. Dahlen, M. Osborn, N. Okulan, W. Foreman, and A. Chand, "Tip characterization and surface reconstruction of complex structures with critical dimension atomic force microscopy," *American Vacuum Society*, no. 2297, 2005.

[105] J. S. Villarrubia, "Algorithms for Scanned Probe Microscope Image Simulation, Surface Reconstruction and Tip Estimation," *Journal of Research of NIST*, vol. 102, no. 4, 1997.

[106] T. Machleidt, E. Sparrer, D. Kapusi, and K-H Franke, "Deconvolution of Kelvin probe force microscopy - methology and application," *Meas. Sci. Technol.*, vol. 20, p. 084017(6 S.), 2009.

[107] Computer Vision Center "Argus" Ltd. (2014, Oct.) calib3d. Camera Calibration and 3D Reconstruction. [Online]. http://docs.opencv.org/modules/calib3d/doc/calib3d.html

[108] B Jähne, *Digitale Bildverarbeitung*, 6th ed. Berlin: Springer, 2005.

[109] Computer Vision Center "Argus" Ltd. (2014, Oct.) openCV. [Online]. http://opencv.org/

[110] H. A. Mirau, "Interferometer," US Patent 2612074, Mar. 27, 1950.

[111] Allied Vision Technology. (2012, Oct.) Prosilica GC 680. [Online]. http://www.alliedvisiontec.com/de/produkte/kameras/gigabit-ethernet/prosilica-ge/ge680.html

[112] S. Trittler, *Processing of Interferometric Data*. Heidelberg: Ruprecht-Karls-Universität, 2007.

[113] T. Seiffert, *Verfahren zur schnellen Signalaufnahme in der Weißlichtinterferometrie*. Erlangen: Friedrich-Alexander-Universität Erlangen-Nürnberg, 2005.

[114] J. C. Wyant, "White Light Interferometry," *Proc. of SPIE*, no. 4737, 07/2002.

[115] O Date, *Erweiterung eines Leica Mikroskops zur Auswertung von Weißlichtinterferogrammen für die Bestimmung dreidimensionaler Oberflächentopographien von Mikro- und Nanostrukturen*, TU Ilmenau - FG Graphische Datenverarbeitung, Ed. Ilmenau: Diplomarbeit, 2010.

[116] D Kapusi, T Machleidt, K-H Franke, E Manske, and R Jahn, "Measuring large areas by white light interferometry at the nanopositioning and nanomeasuring machine (NPMM)," in *52. Internationales Wissenschaftliches Kolloquium*, Ilmenau, 10-13 September 2007.

[117] T Machleidt, E Sparrer, E Manske, D Kapusi, and K-H Franke, "Area-based optical 2.5D sensors of a nanopositioning and nanomeasuring machine," *Meas. Sci. and Technol.*, vol. 23, no. 7, p. 074010, Juli 2012.

[118] T Machleidt, D Kollhoff, O Date, D Kapusi, and R Franke, K-H Nestler, "Nutzung von Farbkameras für die 3D-Oberflächenmessung mittels Weißlichtinterferometrie," in *16. GMA/IGT Fachtagung Sensoren und Messsysteme*, Nürnberg, 2012, pp. 482-489.

[119] T Machleidt et al., "Untersuchung des Einsatzes von Farbkameras zur 3D-Oberflächenmessung mittels Weißlichtinterferometrie," in *17. Workshop Farbbildverarbeitung*, Konstanz, 29-30.9.2011.

[120] M. Hißmann, *Bayesian Estimation for White Light Interferometry*. Heidelberg: Ruprecht-Karls-Universität Heidelberg, 2005.

[121] International Organization for Standardisation. (2007, September) Programming languages - C. [Online]. http://www.open-std.org/JTC1/SC22/WG14/www/docs/n1256.pdf

[122] E Sparrer, T Machleidt, T Hausotte, K-H Franke, and E Manske, "A framework for using optical sensors in nanomeasuring machines over I++/DME," *Meas. Sci. Technol.*, vol. 32, no. 7, p. 07013, July 2012.

[123] E Sparrer, T Machleidt, T Hausotte, E Manske, and K-H Franke, "A Framework for optical Sensors utilizing I++DME On Nanopositioning And Nanomeasuring Machines," in *56. Internationales Wissenschaftliches Kolloquium*, Ilmenau, 12-16. September 2011.

[124] M. Neugebauer and U. Neuschaefer-Ruber, "A new micro artefact for testing of optical and tactile sensors," *Proc. of 5th EUSPEN International Conference*, vol. Vol. 1, 2005.

[125] T Williamson, *A High-Performance Stereo Vision System for Ostacle Detection*, Carnegie Mellon University, Ed. Pitsburgh, 1998.

[126] R Wiesendanger, *Scanning Probe Microscopy and Spectroscopy: Analytica Methods*. Cambridge: Springer, 1998.

[127] W. van Vliet, *Development of a fast mechanical probe for coordinate measuring machines*., 1996.

[128] D Vehar, *Metrische 3D-Rekonstruktion von natürlichen Szenen durch Bildanalyse binokularer Kameras*, Diplomarbeit, Ed. Ilmenau: TU-Ilmenau, 2011.

[129] R. Tutsch et al., "Taktil-optischer 3D-Taster zur Messung von Mikrobauteilen," *tm - Technisches Messen*, vol. 77, no. 2, 2010.

[130] K. Takamasu, R. Furutani, and S. Ozono, "Development of nano-CMM (Coordinate Measuring Machine with Nanometer Resolution)," *Proc. of IMEKO, XIV*, 1997.

[131] E Sparrer, T Machleidt, T Hausotte, E Manske, and K-H Franke, "Integration of CMM software standards for nanopositioning and nanomeasuring machines," in *Proc. of SPIE*, vol. Vol 8031 Pt. 2, Orlando, 2011, p. 8031 2G.

[132] H. Schwenke, U. Neuschaefer-Rube, and T. Pfeifer, "Optical Methods for Dimensional Metrology in Production Engineering," , vol. 2, 2002, pp. 685-699.

[133] O Schreer, *Stereoanalyse und Bildsynthese*. Berlin: Springer-Verlag, 2005.

[134] W. Rauh, "Präzision mit glänzender Faser - Messen von Mikrostrukturen und -bauteilen," *Mikroproduktion*, no. 1/2005, 2005.

[135] W. O. Pril, *Development of high precision mechanical probes for coordinate measuring machines*., 2002. [Online]. http://alexandria.tue.nl/extra2/200213833.pdf

[136] M Nonnenmacher, M P O'Boyle, and H K Wickramainghe, "Kelvin force," *Appl. Phys. Ltr.*, vol. 25, no. 58, pp. 2921-3, June 1991.

[137] U. Neuschaefer-Rube and M. Wissmann, "Taktil-optischer 3D-Mikrotaster - Anordnung und

Messmethode," *PTB-Mitteilungen*, vol. 4, no. 117, 2007.

[138] U. Neuschaefer-Rube and M. Wissmann, "Tactile-optical 3D sensor applying image processing," in *Proc. of SPIE*, vol. 7239, 2009, p. 72390H.

[139] U. Neuschaefer-Rube, K. Wendt, and W. Ehring, "Optische Sensoren für die Koordinatenmesstechnik - Prinzipien und Prüfung," *PTB-Mitteilung*, vol. 117, no. 4, pp. 380-389, 2007.

[140] H.J. Neumann, *Präzisionsmesstechnik in der Fertigung mit Koordinatenmessgeräten*. Renningen: Expert Verlag, 2005.

[141] F. Meli et al., "High precision, low-force 3D touch probe for measurements on small objects," in *EUSPEN Int. Topical Conf.*, Aachen, 2003, pp. 411–414.

[142] Y Martin and H K Wickramasinghe, "Method for imaging sidewalls by atomic force microscopy," *Appl. Phys. Lett.*, vol. 2498, no. 64, 1994.

[143] S N Magonov and W Myung-Hwan, *Surface Analysis with STM and AFM*. Weinheim: Wiley-VCH, 1996.

[144] T Machleidt et al., "Using defined structures in very thin foils for characterizing AFM tips," *Ultramicroscopy*, vol. 107, no. 10-11, pp. 1086-90, Oct. 2007.

[145] W. P. Linnik, "A Simple Interferometer for the Investigation of Optical Systems," *Applied Optics*, vol. 18, no. 12, pp. 2010-2, 1979.

[146] A Küng and F Meli, "Self calibration method for 3D roundness of spheres using an ultraprecision coordinate measuring machine," in *Proc. 5th Euspen Int. Conf.*, Montpellier, Frankreich, 2005, pp. 193-6.

[147] D. Kollhof and K.-H. Franke, "VIP-Toolkit - ein anspruchsvolles, modulares Programmpaket zum Rapid-Prototyping von Bildverarbeitungsanwendungen," , Vortrag im Rahmen der Workshopreihe: "Automation durch integrierte Bildverarbeitung - Workshop 1: Flexible Anwendung modularer Inspektionssysteme", 2003.

[148] D. Kapusi, T. Machleidt, and K.-H. Franke, "Verbundprojekt Hochpräzise optoelektronische Messsysteme für die Präzisionsfertigung (HOMS) : Teilprojekt 3 : Messwerterfassung und -auswertung zur Vermessung von Lehren," Projektabschlußbericht, Fachgebiet Graphische Datenverarbeitung, Ilmenau, Förderkennzeichen BMBF 03/2913 A - Verbund-Nr. 01031832, 06/2006.

[149] G. Jäger et al., "Nanometrology – Nanopositioning- and Nanomeasuring Machine," *Materials Science Forum*, no. 505-507, pp. 7-12, 2006.

[150] T. Hausotte, *Nanopositionier- und Nanomessmaschinen: Geräte für hochpräzise makro- bis nanoskalige Oberflächen- und Koordinatenmessungen*, 1st ed.: Pro Business, 2011.

[151] T. Hausotte, B. Percle, and G. Jäger, "Advanced three-dimensional scan methods in the nanopositioning and nanomeasuring machine," *Meas. Sci. Technol.*, no. 084004, 2009.

[152] D. Grisby. (2009, September) Free High Performance ORB. [Online]. http://omniorb.sourceforge.net/

[153] Nanotools GmbH. (2012) Nanotools Scanning Probes.

[154] E. Gamma, R. Helm, R. Johnson, and J Vlissides, *Entwurfsmuster - Elemente wiederverwendbarer objektorientierter Software*. München: Addison Wesley Verlag, 2004.

[155] W. Dutschke, *Fertigungsmesstechnik*. Stuttgart: B.G. Teubner, 2002.

[156] R. Christoph and H.J. Neumann, *Multisensor-Koordinatenmesstechnik*. München: sv corporate media, 2006.

[157] H. Bosse and G. Wilkening, "Dimensionelle Nanometrologie in der PTB - eine Übersicht," *Technisches Messen*, vol. 73, pp. 4-18, 2006.

[158] B Bhushan, *Springer Handbool of Nanotechology*, 3rd ed.: Springer, 2012.

[159] Simonsen, K. (2010, Oktober) Welcome to the official home of JTC1/SC22/WG21 - The C++ Standards Committee. [Online]. http://www.open-std.org/JTC1/SC22/WG21/

[160] Image Metrology A/S. (2011, April) Scanning Probe Image Processor. [Online]. http://www.imagemet.com/index.php?main=products

[161] International Organization for Standardisation, *Programming language C++. September 1998*. Genf: International Organization for Standardisation, 1998.

[162] Hexagon Metrology, "Paperless Quality," *Fabricating & Metalworking*, 2007.

[163] Deutsches Institut für Normung e. V., *Optoelektronische Abstands-, Profil- und Formmessung*. Berlin: Beuth, 08/2000.

[164] XspectSolutions Inc. (2005, Mai) openDMIS. [Online]. http://www.opendmis.com/Brochure/DmistifyingDMIS.pdf

[165] Object Management Group. (2010, März) Object Management Group. [Online]. www.omg.de

[166] National Institute for Standards and Technology. (2009, Juli) NIST Standards Activities in Dimensional Metrology. [Online]. http://www.isd.mel.nist.gov/projects/metrology_interoperability/NISTactivities.htm#utilities

[167] Renishaw. (2010, Januar) MODUS. [Online]. http://www.renishaw.de/de/10488.aspx

[168] Hexagon Metrology SpA. (2012) Leitz Infinity. [Online]. http://www.leitz-metrology.com/deu/leitz-infinity_541.htm

[169] Deutsches Institut für Normung e. V., *Industrial automation systems and integration - Physical device control - Dimensional Measuring Interface Standard (DMIS)*. Berlin: Beuth, 2011.

[170] Wenzel Präzisions GmbH, "I++DME: Ziele, Konzepte und deren Umsetzung in Produkten," *VDI-Z Integrierte Produktion*, no. 10, p. 2, 2006. [Online]. http://www.wenzel-cmm.com/wpgroup/08_pdf/de/wenzel-i-plus.pdf

[171] Deutsches Institut für Normung e.V., *Geometrische Produktspezifikation - Annahmeprüfung und Bestätigungsprüfung für Koordinatenmessgeräte*. Berlin: Beuth, 2010.

[172] VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, *Genauigkeit von Koordinatenmessgeräten - Kenngrößen und deren Prüfung - Koordinatenmessgeräte mit optischer Antastung*. Düsseldorf: VDI Verlag, 2007.

[173] International Organization for Standardisation, *Evaluation of measurement data - Guide to the expression of uncertainty in measurement*. Genf: International Organization for Standardisation, 2008.

[174] Microsoft Corporation. (2010, Dezember) Dynamic-Link Libraries. [Online]. http://msdn.microsoft.com/en-us/library/ms682589.aspx

[175] Inspec Software Corp. (2007, November) Dimensional Measurement Interface Standard. [Online]. www.dmis.com

[176] Hexagon Metrology, "Case Study Robert Bosch GmbH Kalibrieren mit der Königsklasse," Robert Bosch GmbH, Fallstudie 2011.