Entwicklung und Implementierung eines Planungssystems zur robotergestützten Laserosteotomie

Diplomarbeit

Martin Lohmann

DLR

Institut für Robotik und Mechatronik	BJ.: 2008
	IB.Nr.: 515-11-03

DIPLOMARBEIT

ENTWICKLUNG UND IMPLEMENTIERUNG EINES PLANUNGSSYSTEMS ZUR ROBOTERGESTÜTZTEN LASEROSTEOTOMIE

Freigabe:	Der Bearbeiter:	Unterschriften
	Martin Lohmann	
	Betreuer:	
	Rainer Konietschke	
	Der Institutsdirektor	
	Prof. Dr. G. Hirzinger	

Dieser Bericht enthält 119 Seiten, 68 Abbildungen und 10 Tabellen

Ort: Oberpfaffenhofen	Datum:	Bearbeiter:	Zeichen:

Aufgabenstellungen für eine Diplomarbeit

Martin Lohmann

Titel der Arbeit: Entwicklung und Implementierung eines Planungssystems zur robotergestützten Laserosteotomie

Kategorie: ☒ Diplomarbeit

Dauer: 6 Monat(e)

Studienrichtung: Maschinenbau, Mechatronik, Informatik

Aufgabenstellung: **Hintergrund.** Der Einsatz der Lasertechnologie zur Trennung von Körpergewebe ist z.B. aus der Zahnheilkunde bekannt. Entsprechende Laser können auch zum Trennen von Knochengewebe (Osteotomie) eingesetzt werden. Im Vergleich zu herkömmlichen Methoden (z.B. Knochensäge oder -meißel) erhofft man sich damit folgende Vorteile:

- Höhere Genauigkeit und ästhetische Vorteile durch präoperative Planung und (hoch)genaue Umsetzung
- Geringere Schnittbreite und damit geringerer Verlust an gesundem Knochengewebe
- Beliebige Patienten- und Fall-spezifische Extrusions-Schnittformen, u.a. erzielen geschwungene Strukturen höhere Stabilität
- Geringeres Trauma da kein physischer Kontakt von Instrument und Patient
- Nahezu reaktionskraftfreies Führen des Instruments
- Exaktes Schneiden gemäß präoperativ geplanter Daten, daher kann z.B. ein Implantat vorab gefertigt werden

Die Anwendung von Lasertechnologie stellt allerdings auch einige bisher weitgehend ungelöste Aufgaben:

- Begrenzung des thermischen Energieeintrags in das Knochengewebe, sodass Nekrosen vermieden werden
- Modellierung des Schneidvorgangs, Bestimmung der Abhängigkeit des Abtragsvolumens von der Knochendichte
- Entwicklung einer aktiven Schnitttiefenmessung
- Optimierung des Schneidprozesses, da der Laser ein vergleichsweise geringes Volumen pro Zeit abträgt im Vergleich zu konventionellen Instrumenten
- Exakte Realisierung einer geplanten Schnittgeometrie lässt sich durch manuelles Führen des Lasers nur begrenzt erreichen

Am Institut für Robotik und Mechatronik des DLR wird ein System zur robotergestützen Laserosteotomie aufgebaut. Dabei kommt der am Institut entwickelte medizinische Leichbauroboter MIRO *[1]* (s. Abb. 1) zum Einsatz.

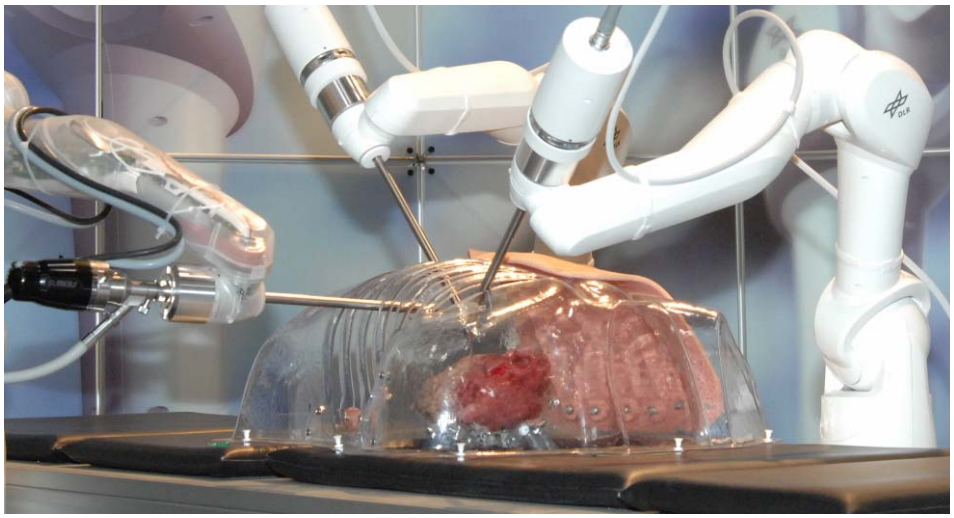


Abbildung 1: 3 MIRO-Roboter in einer minimal invasiven Konfiguration

Ziel dieser Arbeit: Ein wesentlicher Vorteil des Einsatzes eines Roboters zur Führung des Schneidlasers besteht darin, dass präoperativ geplante Volumina exakt intraoperativ abgetragen werden können. Dazu ist es nötig, eine präoperative Planung durchzuführen, welche sowohl die patientenspezifische Anatomie des Patienten, als auch die Kinematik des Roboters und ein Abtragsmodell des Schneidlasers berücksichtigt. Als Ergebnis der präoperativen Planung entsteht die formale Beschreibung eines optimierten Ablaufs der Operation. Intraoperativ soll die geplante Trajektorie entsprechend der zur Verfügung stehenden Sensordaten adaptierbar sein, um auf Unterschiede zwischen Planung und Realität eingehen zu können.

Ziel dieser Arbeit ist es, ein Planungssystem zu entwickeln, welches es ermöglicht, präoperativ den Eingriff der Laserosteotomie zu planen. Die Ergebnisse der Planung sollen dann intraoperativ adaptierbar sein an Sensordaten wie zum Beispiel die Messung der aktuellen Schnitttiefe.

Aufgabenstellung. Zur Bearbeitung des Themas können folgende Arbeitsschritte identifiziert werden:

1. Recherche in Frage kommender Operation; Aufzeichnung des genauen Ablaufs der herkömmlichen Methode von Beginn an (in Kontakt mit Ärzten).
2. Erweiterung der herkömmlichen Methode: Skizzieren des möglichen Ablaufs bei Roboterunterstützung. Im Bereich der Planung: an welchen Stellen müssen zusätzliche Schritte eingeführt werden?
3. Identifizierung der Schnittstellen zur herkömmlichen Methode und zum Roboter- und Lasersystem. Definition neu zu erstellender Module.
4. Anpassung der bestehenden Simulation des realen Robotersystems an die spezielle Aufgabe des Laserschneidens, um eine Testumgebung der neu zu entwickelnden Module aufzubauen.
5. Implementierung des Planungssystems. Funktionen könnten u.a. sein:
    - Einzeichnen von Schnitten in 3D-Modelle oder in CT-Daten, daraus Erstellen von 3D-Volumina
    - Setupoptimierung: Platzierung von Roboter und Laser, sodass gesamtes Volumen von Laser erreicht wird (insbesondere Einbezug von Überschneidungen)

- Erstellen eines (adaptierbaren) Schneidprogramms bei Annahme eines bestimmen Laser-Knochen-Abtragmodells
- VR-Simulation des Abtragens
- Intraoperativ wird der tatsächliche Abtrag gemessen. Effizienter Vergleich dieser Messung mit dem Abtrag laut Planung und daraufhin Adaption der Planung

Die Dokumentation der Arbeit erfolgt schritthaltend mit der Bearbeitung. Nach ca. 3 Monaten wird ein Zwischenbericht angefertigt, der dann auch in die abschliessende Dokumentation der Diplomarbeit übernommen werden kann. Die Arbeit könnte im August 2010 beginnen und ist auf 6 Monate ausgelegt.

Literatur:

[1] Hagn, U., Nickl, M., Jörg, S., Tobergte, A., Kübler, B., Passig, G., Gröger, M., Fröhlich, F., Seibold, U., Konietschke, R., Le-Tien, L., Albu-Schäffer, A., Grebenstein, M., Ortmaier, T., und Hirzinger, G. (2008), "DLR MIROSURGE – towards versatility in surgical robotics", in Proceedings of curac.08, 2008,7. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie e.V., Leipzig, ISBN 978-3-00-025798-8, pp. 143-146

Link zum Projekt: http://www.dlr.de/rm/desktopdefault.aspx/tabid-3835/6288_read-9047/

Geforderte/ gewünschte Fähigkeiten:

- Studienrichtung Maschinenbau, Mechatronik oder Informatik
- Grundkenntnisse in Robotik
- Grundkenntnisse in Programmierung mit C/C++
- Grundkenntnisse MatLAB/Simulink
- Freude an interdisziplinären Aufgabenstellungen

Kontakt:

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Robotik und Mechatronik

Dr.-Ing. Rainer Konietschke

Dipl.-Ing Florian Fröhlich

Münchnerstrasse 20

D-82234 Wessling

Telefon: ++49 (0) 81 53 / 28 - 24 98
Telefax: ++49 (0) 81 53 / 28 - 11 34
rainer.konietschke@dlr.de

Diplomarbeit

Entwicklung und Implementierung eines Planungssystems zur robotergestützten Laserosteotomie

Martin Lohmann

31. März 2011

Betreuer:	Institut:
Prof. Dr.-Ing. Gunther Reinhart Prof. Dr.-Ing. Michael F. Zäh Dipl.-Ing. Jens Hatwig	Institut für Werkzeugmaschinen und Betriebswissenschaften Technische Universität München
Prof. Dr.-Ing. Gerhard Hirzinger Dr.- Ing. Rainer Konietschke	Institut für Robotik und Mechatronik Deutsches Zentrum für Luft- und Raumfahrt

Vorwort

Die Diplomarbeit ist im Rahmen meines Studiums für Maschinenwesen am Institut für Robotik und Mechatronik des Deutschen Zentrums für Luft- und Raumfahrt entstanden und wurde vom Institut für Werkzeugmaschinen und Betriebswissenschaften an der Technischen Universität München betreut. Für die Ermöglichung dieser Arbeit bedanke ich mich herzlich bei der Leitung der Institute Professor Dr.-Ing. Gerhard Hirzinger, Professor Dr.-Ing. Gunther Reinhart sowie Professor Dr.-Ing. Michael Zäh. Darüber hinaus möchte ich mich bei meinen beiden Betreuern Dr.-Ing. Rainer Konietschke und Dipl.-Ing. Jens Hatwig bedanken, welche durch ihre fachliche und organisatorische Unterstützung maßgeblich am Erfolg dieser Arbeit beteiligt waren. Für die fortwährende moralische Unterstützung bedanke ich mich bei meiner Freundin Verena.

Inhaltsverzeichnis

Vorwort **i**

Abkürzungen und Symbole **v**

1 Einleitung **1**

2 Stand der Technik und Grundlagen **3**

2.1 Robotergestützte Laserosteotomie 3

2.1.1 Aufbau des Knochens 4

2.1.2 Grundlagen der Lasertechnik 6

2.1.3 Einsatz von Robotik 10

2.2 Planung für die robotergestützte Laserosteotomie 15

2.2.1 Planungsmethoden Osteotomie 16

2.2.2 Planungsmethoden Laserosteotomie 17

3 Motivation **21**

4 Aufbau des Planungssystems **23**

4.1 Gesamtsystem 23

4.1.1 Statische Systembeschreibung 23

4.1.2 Verwendete Komponenten 26

4.1.3 Dynamische Systembeschreibung 29

4.2 Vorbereitung der anatomischen Bilddaten 31

4.3 Einrichten der Operationsumgebung 33

4.4 Volumenmodellierung und Schnittplanung 35
4.4.1 Volumenmodellierung 35
4.4.2 Schnittplanung 39
4.5 Modellierung des Laserabtrags 45
4.5.1 Physikalische Vorgänge 45
4.5.2 Mathematisches Modell 47
4.5.3 Implementierung des Modells 51
4.6 Präoperative Abtragssimulation und -visualisierung 60
4.7 Intraoperative Abtragsberechnung und -visualisierung 65

5 Experimente **71**
5.1 Experiment zur Beurteilung des Abtragsmodells 72
5.2 Experiment zur Beurteilung des Arbeitsablaufs 84

6 Diskussion und Ausblick **87**

Literaturverzeichnis **89**

Anhang **95**

A Softwaredokumentation **97**

B Verwendete Dateiformate **107**

Eidesstattliche Erklärung **119**

Abkürzungen und Symbole

Selten benutzte Abkürzungen und Symbole werden ausschließlich im Text erläutert.
Fett gedruckte Symbole repräsentieren Vektoren und Matrizen.

Abkürzungen

TEM_{lp}	Transversal-Elektromagnetische Mode
CT	Computertomographie
MRT	Magnetresonanztomographie
DICOM	Digital Imaging and Communications in Medicine
TCP	Tool Center Point
DH-Parameter	Denavit Hartenberg Parameter
GUI	Grafical User Interface

Symbole

λ	Wellenlänge	$[\mu m]$
w_0	Fokusradius des Laserstrahls	$[mm]$
z_r	Rayleighlänge des Laserstrahls	$[mm]$
P_E	Pulsenergie	$[mJ]$
Φ	Pulsenergiedichte	$[J/cm^2]$
f_p	Pulsfrequenz	$[Hz]$
t_p	Pulsweite bzw. Pulsdauer	$[\mu s]$
t_{th}	Thermische Relaxationszeit	$[\mu s]$
P	Leistung	$[W]$
M_Z	Einzelpulsmodell Zylindrischer Krater	$[-]$
M_G	Einzelpulsmodell Gaußscher Krater	$[-]$
S_{PP}	Simulationsart Punktpuls	$[-]$
S_{LP}	Simulationsart Linienpuls	$[-]$
$\Delta h(r,h)$	Schnitttiefe eines Einzelpulsabtrags	$[mm]$

h	Kratertiefe (Summe der Schnitttiefen)	$[mm]$
Φ_S	Ablations- oder Abtragsschwelle eines Materials	$[J/cm^2]$
α	Absorptionskoeffizient eines Materials	$[cm^{-1}]$
μ	Abschwächungskoeffizient eines Materials	$[cm^{-1}]$
${}^{y}\boldsymbol{T}_{x}$	Homogene 4x4-Matrix zur Beschreibung von Lage und Orientierung des Punkts x im Raum y	$[-]$
${}^{y}\boldsymbol{p}$	Translatorischer Anteil der Matrix ${}^{y}\boldsymbol{T}_{x}$	$[-]$

Kapitel 1

Einleitung

Das Trennen von Hartgewebe mithilfe der Lasertechnologie ist aus der Zahnheilkunde bekannt. In ähnlicher Weise können Laser zum Schneiden oder Abtragen von Knochenmaterial verwendet werden (Laserosteotomie). Im Vergleich zu herkömmlichen Methoden wie Sägen oder Fräsen erhofft man sich dabei u.a. eine verbesserte Schnittgenauigkeit und geringere Schnittbreite, sowie eine höhere Flexibilität in der Schnittführung. Diese Präzision könnte es z.B. ermöglichen, präoperativ geplante Implantate ohne Nacharbeit exakt und direkt einzupassen. Darüber hinaus wäre die Anfertigung komplexerer Schnittgeometrien wie geschwungene Linien denkbar. Durch die resultierende vergrößerte Schnittfläche könnten Implantate besser einheilen und Knochenflächen besser zusammenwachsen. Sogar so genannte Self Lock Mechanismen durch das Lasern von schwalbenschwanzförmigen Grenzflächen, wie in Abbildung 1.1 dargestellt ist, sind denkbar.

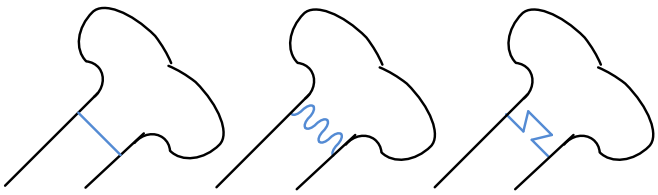


Abbildung 1.1: Links: Übliche Schnittform bei einer Osteotomie mit normalem Schnittwerkzeug. Mitte und Rechts: Mögliche Schnittformen bei der Laserosteotomie.

Eine Anwendung könnte zum Beispiel die Korrekturosteotomie von Femurfehlstellungen (Fehlstellungen des Oberschenkelknochens) sein. Bei dieser wird ein Keil aus dem Femur geschnitten, wonach Ober- und Unterteil durch Kippung und Torsion in korrigierter Lage wieder zusammengefügt werden. Zur postoperativen Versorgung werden oft Platten oder Drähte eingesetzt um die Knochen zusammenzuhalten. Durch Self Lock Mechanismen oder geschwungene Linien könnte dieser Heilungsprozess beschleunigt werden.

Während die Anwendung des Lasers in einigen Bereichen der Medizin schon längst zum Stand der Technik gehört, ist die Laserosteotomie jedoch bisher nur Thema an wissenschaftlichen Instituten und Universitäten. Grund dafür sind die unüberwundenen Probleme, wie z.B. die Entwicklung von Nekrosen durch überhöhte thermische Einwirkung auf die zu schneidende

Stelle. Nekrosen können u.a. entstehen, wenn der Laserstrahl bezüglich Einfallswinkel und Fokusabstand nicht exakt zur Knochenoberfläche geführt wird. Dieser Nachteil könnte durch den Einsatz eines Roboters überwunden werden. Ein wesentlicher Vorteil des Einsatzes eines Roboters zur Führung des Schneidlasers besteht darin, dass präoperativ geplante Volumina intraoperativ exakt abgetragen werden könnten. Dazu ist eine präoperative Planung notwendig, welche sowohl die patientenspezifische Anatomie als auch die Roboterkinematik und ein Abtragsmodell des Lasers berücksichtigt. Um auf Unterschiede zwischen Realität und Planung eingehen zu können, ist die präoperative Planung im Idealfall durch Sensordaten, z.B. zur Abtragstiefenmessung, anpassbar.

Im Folgenden wird ein Ansatz für ein solches Planungssystem vorgestellt. Dazu werden zuerst in Kapitel 2 notwendige Grundlagen und bisher auf diesem Gebiet durchgeführte Arbeiten erläutert. Danach werden die genauen Ziele in Kapitel 3 formuliert und das erarbeitete Konzept in Kapitel 4 dargestellt. Zur Evaluierung des Systems durchgeführte Experimente werden in Kapitel 5 besprochen, in Kapitel 6 wird das Erreichte bewertet und es werden weiterführende Möglichkeiten diskutiert.

Kapitel 2

Stand der Technik und Grundlagen

2.1 Robotergestützte Laserosteotomie

Wie in Abbildung 2.1 zu sehen ist, besteht das System für eine robotergestützte Laserosteotomie aus einem Roboter und einer Laserquelle. Der Laserstrahl wird über einen Scankopf oder direkt über einen Laserendeffektor freigegeben. Beim Einsatz eines Scankopfes wird der Laserstrahl durch eine separate Steuereinheit im Scankopf ausgelenkt, der Roboter hält dabei nur den Scankopf selbst. Der Strahl kann so sehr genau und hochdynamisch positioniert werden, da er im Scankopf über Spiegel ausgelenkt wird, deren Gewicht sehr gering ist. Jedoch ist der Arbeitsbereich des Scankopfes meist klein, so dass vom Roboter gegebenenfalls verschiedene Positionen des Scankopfes angefahren werden müssen, um einen größeren Arbeitsbereich abzudecken. Die Genauigkeitsanforderungen an den Roboter sind bei dieser Methode geringer als wenn er direkt einen Laserendeffektor führt. Bei der Laserosteotomie ohne Scankopf wird der Laserstrahl direkt über einen Laserendeffektor freigegeben. Ähnlich einer manuellen Laserosteotomie, bei der der Arzt den Endeffektor entlang der gewünschten Schnittlinie über die zu entfernende Knochenstruktur führt, wird diese Bewegung hierbei vom Roboterarm ausgeführt. Die Genauigkeitsanforderungen an den Roboter sind entsprechend höher, jedoch bringt diese Methode eine vereinfachte steuerungstechnische Umsetzung mit sich. In der vorliegenden Arbeit wird ein Ansatz verfolgt, den Laser ohne Scankopf frei zu führen.

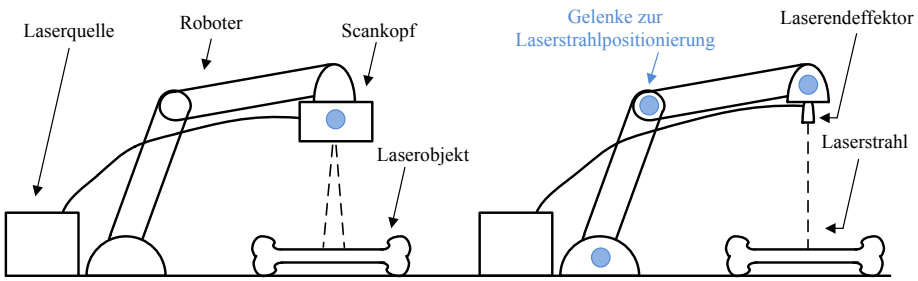


Abbildung 2.1: Mögliche Aufbauten eines Systems zur Laserosteotomie. Links: Mit Scankopf. Rechts: Ohne Scankopf.

Im Folgenden werden die Grundlagen erläutert, welche zum Verständnis der Prozesse während einer robotergestützten Laserosteotomie notwendig sind. Zudem werden Arbeiten aus der Literatur zur Laserosteotomie zusammengefasst. In Abschnitt 2.1.1 und 2.1.2 werden dazu zuerst der Aufbau des Knochens sowie Grundlagen zur Lasertechnik erklärt. Darauf folgend wird in Abschnitt 2.1.3 ein kurzer Einblick in robotergestützte Laserbearbeitungsmethoden in der Industrie und in die chirurgische Robotik gegeben. Außerdem werden aktuelle Forschungsarbeiten zur robotergestützten Laserosteotomie besprochen.

2.1.1 Aufbau des Knochens

Morphologie des Knochens Neben dem Zahnmaterial ist Knochengewebe das härteste Material im menschlichen Körper. Knochen ist widerstandsfähig gegen Druck, Zug, Torsion und Biegung [39]. In Abbildung 2.2 ist ein Femur (Oberschenkelknochen) dargestellt.

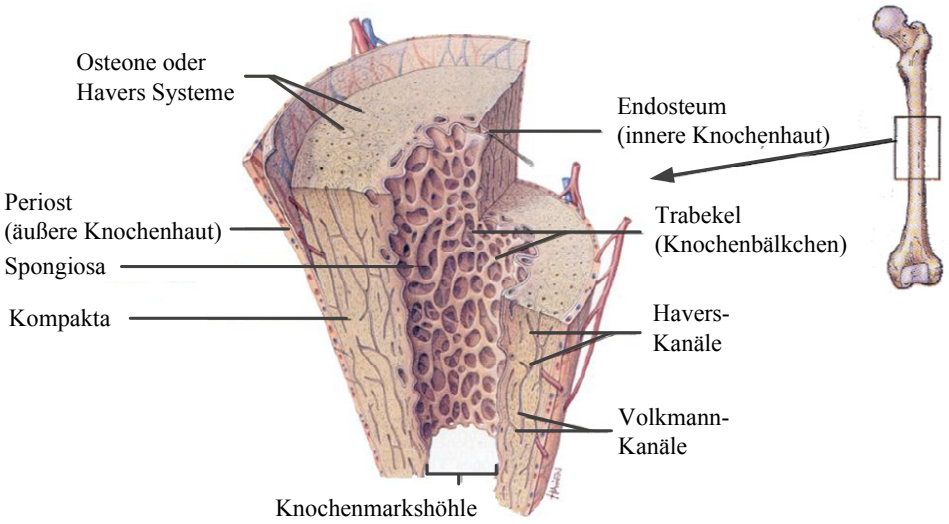


Abbildung 2.2: Anatomischer Aufbau eines Femurknochens, entnommen aus [30].

Der Femur gehört zu den Röhrenknochen, denen u.a. noch der Humerus (Oberarmknochen) und die Fibula (Wadenbein) angehören. Sie alle bestehen aus zwei Knochenenden und einem langen Schaft, sowie einer langen Markhöhle, die sich durch den ganzen Knochen zieht. Operationen am Femur, wie der Gelenkersatz mittels Hüftgelenksendoprothese oder die Femurfehlstellungskorrektur, haben große Relevanz in der Patientenversorgung der Industriestaaten [54]. Auch in der Forschung spielen Operationen am Femur eine große Rolle, was die hohe Anzahl der Treffer in den Literaturdatenbanken zeigt (Medline > 57000, Januar 20011). Außerdem stellt der Femurknochen mit bis zu $3\ cm$ Schaftdurchmesser [42] den größten Knochen des Menschen und damit auch die höchsten Anforderungen bezüglich des Laserschneideprozesses dar. Aus diesen Gründen soll hier näher auf ihn eingegangen und er auch im weiteren Verlauf dieser Arbeit zur beispielhaften Laserosteotomie verwendet werden.

Der morphologische Aufbau ist bei den meisten der 206 Knochen im menschlichen Körper sehr ähnlich. Der Knochen ist mit der Knochenhaut, dem sogenannten Periost überzogen, danach folgt eine äußere Knochenstruktur, welche als Kompakta bezeichnet wird. Das Knocheninnere

ist durch eine schwammartige Struktur, die Spongiosa, bestimmt. Letztere besteht aus Trabekeln, kleinen Knochenbälkchen, welche in Richtung der Belastungslinien verlaufen. Die Spongiosa bildet Hohlräume in welchen das Knochenmark verläuft. Die Kompakta ist allgemein fester, weshalb sie für die Laserosteotomie eine größere Herausforderung darstellt als die Spongiosa. Darüber hinaus besteht der Schaft des Femur, an dem die meisten Schnitte getätigt werden, hauptsächlich aus Kompakta [30].

Chemischer Aufbau des Knochens Der humane Knochen besteht zu 50 bis 60% aus anorganischen Substanzen, zu 20% aus organischen Substanzen und zu ca. 20% aus Wasser. Hydroxylapatit stellt ungefähr 95% der anorganischen Substanzen dar, der Rest sind Salze wie Natrium. Der organische Teil besteht hauptsächlich aus dem Strukturprotein Kollagen und zu einem kleinen Teil aus anderen Proteinen. Der chemische Aufbau schwankt mit dem Alter und der Knochenart erheblich [14].

Physikalische Eigenschaften des Knochens Bezüglich der physikalischen Eigenschaften sollen die optischen, thermischen, und mechanischen Eigenschaften angesprochen werden, wobei die ersten beiden für die Laserosteotomie die größte Relevanz haben. Ein Laserstrahl wird von Materie reflektiert, transmittiert oder absorbiert. Die Absorption ist für die Laserosteotomie das wichtigste optische Phänomen. Wie viel eines Strahls von einem Material absorbiert wird, bestimmt der Absorptionskoeffizient α $[cm^{-1}]$. Dabei ist die Wellenlänge der Laserstrahlquelle entscheidend, das Absorptionsmaximum von Wasser liegt z.B. bei $2900\ nm$, was im Wellenlängenbereich von Infrarotlasern liegt. Näheres zur Interaktion von Laser und Material wird in Abschnitt 2.1.2 erklärt. Eine Aussage über das thermische Verhalten eines Materials kann u.a. mit der Wärmeleitfähigkeit κ $[W/mK]$ sowie mit der Wärmekapazität C $[J/gK]$ getroffen werden. Im Vergleich zu Materialien, die im industriellen Umfeld mit dem Laser bearbeitet werden, wie Stahl (15-60 W/mK), ist die Wärmeleitfähigkeit von Knochen mit 0.3-0.5 W/mK sehr gering, während die Wärmekapazität mit $1.3\ J/gK$ (bei Stahl ca. $0.5\ J/gK$) vergleichsweise hoch ist. Mechanisch zeichnet sich Knochen durch eine hohe Festigkeit und Flexibilität bei geringem Gewicht aus, was aus einem Zusammenspiel der organischen und anorganischen Komponenten basiert. Ein Knochen kann durch Demineralisierung an Härte verlieren, jedoch Flexibilität und Zähigkeit beibehalten. Im Gegensatz dazu kommt es bei einem Mangel an anorganischen Substanzen zu Knochenbrüchigkeit. In Tabelle 2.1 sind die physikalischen Eigenschaften der Kompakta zusammengefasst.

Parameter	Wert	Einheit
Absorptionskoeffizient ($\lambda = 2900\ nm$) α	3800	$[cm^{-1}]$
Wärmeleitfähigkeit κ	$0,3 - 0,5$	$[W/mK]$
Wärmekapazität C	$1,3$	$[J/gK]$
Dichte ρ	$1,78 - 2,0$	$[g/cm^3]$
Elastizitätsmodul E	$18,5$ (längs); 12 (quer)	$[GPa]$
Max. Zugfestigkeit R_m	195	$[MPa]$
Brinellhärte	24	$[HWB]$

Tabelle 2.1: Physikalische Eigenschaften der Kompakta, entnommen aus [14], [38].

2.1.2 Grundlagen der Lasertechnik

Historische Entwicklung des Lasers Der Begriff Laser steht für *Light amplification by stimulated radiation*. Dies bezeichnete vorerst den physikalischen Effekt der Lichtverstärkung durch stimulierte Emission von Strahlung im Bereich von Röntgen- bis zu Infrarotstrahlung, welcher erstmals 1958 von Charles H. Townes und Arthur L. Schawlow theoretisch gezeigt wurde [48]. Seit 1960 von Theodore Maiman [37] eine entsprechende Strahlenquelle gebaut wurde, beschreibt das Wort Laser allerdings nicht mehr nur den physikalischen Effekt, sondern auch die Strahlenquelle selbst. Schon bald wurden die vielfältigen Anwendungsmöglichkeiten des Lasers ersichtlich, welche heutzutage von Laserschweißen in der Fertigungstechnik, über messtechnische Anwendungen wie beim Laserdopplervibrometer, Anwendungen in der Telekommunikation bis hin zum Einsatz in der Chirurgie reichen. Erste Forschungsschwerpunkte der Laserchirurgie waren die Photocoagulation, eine durch Licht hervorgerufene Blutgerinnung, in der Augenheilkunde, später der Hartgewebeabtrag in der Dentalchirurgie und danach das Laserskalpell in der allgemeinen Chirurgie. Nachdem Zaret 1961 erstmals an Kaninchen eine Laserphotocoagulation zeigte [65], wurde eine solche noch im selben Jahr mithilfe eines von American Optical entwickelten Rubinlaserprototypen von Campbell und Koester [10] an einem Menschen durchgeführt. Bereits 1971 wurde die Photocoagulation mithilfe von Argonlasern an Medizinischen Fakultäten der USA gelehrt [53]. Bezüglich einer Anwendungen zum Hartgewebeabtrag mussten die Forscher zuerst auf höherenergetischere Laser wie den CO_2 Laser warten, welcher 1965 in den Bells Laboratories entwickelt wurde. Vorteile eines Lasereinsatzes in der Medizin sind unter anderem: kontaktloses und damit steriles operieren, höhere Präzision, Gewebeselektivität und gleichzeitige Verödung beim Durchtrennen von Gefäßen.

Physikalische Grundlagen Heutzutage bestehen die verschiedensten Lasertypen, wobei der Großteil gemäß Abbildung 2.3 aufgebaut ist. Da das Laserprinzip ein selbsterregender Oszillator ist, werden ein Verstärkungselement (das Lasermedium), ein Element zur Rückkopplung (der optische Resonator) sowie wegen der Energieerhaltung eine Energiezufuhr (die Pumpquelle) benötigt. Zur Erzeugung eines Laserstrahls sind verschiedene Einzelbedingungen notwendig, welche im Folgenden kurz erläutert werden.

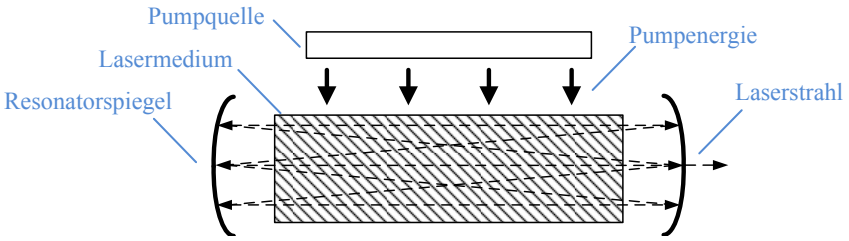


Abbildung 2.3: Grundaufbau eines Lasers.

Die Atome oder Moleküle im Lasermedium besitzen verschiedene Energieniveaus, wobei im Normalzustand mit ansteigender Energie die Population abfällt. Trifft letzteres nicht zu, so wird von Besetzungsinversion gesprochen. Besetzungsinversion ist nur bei bestimmten Materialien möglich und ist abhängig von deren Energieniveaus und Relaxationszeiten. Es gibt Laser mit

gasförmigen, flüssigen sowie festen Lasermedien. Nach dem Lasermedium richtet sich auch der Pumpmechanismus. Bei einem Helium-Neon Gaslaser wird beispielsweise mit einer Gleichstromquelle elektrisch gepumpt, bei einem Nd:YAG Festkörperlaser kann über Blitzlampen optisch gepumpt werden, darüber hinaus gibt es chemische Pumpmechanismen. Zum Hervorrufen einer Besetzungsinversion werden möglichst viele Atome oder Moleküle im Lasermedium in einen über dem Energiegrundniveau liegenden Energiezustand angeregt. Trifft nun ein energetisch passendes Lichtquant auf ein angeregtes Atom oder Molekül, so kann ein zweites Lichtquant ausgelöst werden, welches kohärent ist, d.h. die gleiche Richtung, Wellenlänge, Frequenz, Phase und Polarisation besitzt. Das Atom oder Molekül fällt dabei auf ein niedrigeres Energieniveau zurück. Ein Lichtquant, welches durch ein besetzungsinvertiertes Lasermedium geleitet wird, wird kontinuierlich verstärkt, was dem Laser seinen Namen *Light amplification by stimulated radiation* gibt. Dieser Prozess benötigt als Startbedingung einen Zustand in dem mehr Lichtquanten spontan emittiert werden, als durch Verluste verloren gehen. Um bei begrenzter Baugröße den Laserstrahl optimal zu verstärken wird er an Resonatorspiegeln, die das Lasermedium seitlich begrenzen, reflektiert. Einer der Resonatorspiegel ist an einer bestimmter Stelle transmittierend um den Strahl nach vollendeter Verstärkung freizugeben. Die Spiegel sind oft konkav und konfokal um eine stabile Strahlenausbreitung zu gewährleisten.

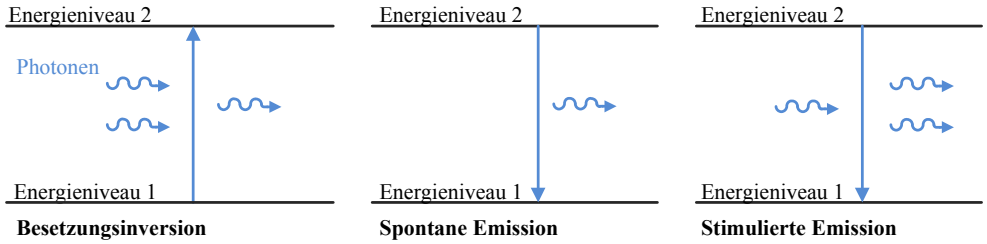


Abbildung 2.4: Grundprinzip eines Lasers. Links: Durch Energiezufuhr (z.B. optisch) wird eine Besetzungsinversion hervorgerufen. Mitte: Spontane Emission als Startbedingung des Laserprozesses. Rechts: Lichtverstärkung durch stimulierte Emission.

Durch stimulierte Emission entstehendes kohärentes Licht unterscheidet sich daher durch folgende physikalischen Eigenschaften von einer konventionellen Lichtquelle, welche nur auf spontaner Emission der Lichtquanten basiert.

- Das Licht ist monochromatisch, d.h. es besitzt eine annähernd konstante Wellenlänge und die Lichtquanten schwingen in Phase. Es ist linear, parallel oder zirkulär polarisierbar.
- Der Laserstrahl besitzt eine geringe Divergenz, da er sich nur unter einem kleinen Öffnungswinkel ausbreitet. Aus dem sehr kleinen Strahldurchmesser resultiert eine hohe Leistungsdichte.
- Über den Strahldurchmesser herrscht eine gaußsche Energieverteilung.
- Die Rayleighlänge des Laserstrahls z_r (Distanz entlang optischer Achse bis zur Verdoppelung der Strahlfläche) mit Wellenlänge λ und Fokusradius w_0 lässt sich wie folgt berechnen:

$$z_r = \frac{\pi \cdot w_0^2}{\lambda} \tag{2.1}$$

- Bei konvokalen Resonatorspiegeln entstehen transversale Moden (vgl. Abbildung 2.5) welche nach ihrer horizontalen (l) und vertikalen (p) Ausrichtung als TEM_{lp}-Moden bezeichnet werden. Meist wird nur die Grundmode TEM_{00} verwendet
- Laser können gepulst oder als Dauerstrichlaser verwendet werden.

Ein Laser wird unter anderem durch seine Wellenlänge λ, seinen Fokusdurchmesser d_0 und seine Leistung P, gepulste Laser außerdem noch durch die Pulsfrequenz f_p und Pulsweite w_p charakterisiert. Genaueres zum Laserprozess kann in einschlägiger Literatur, z.B. Eichler et al. [15] nachgeschlagen werden.

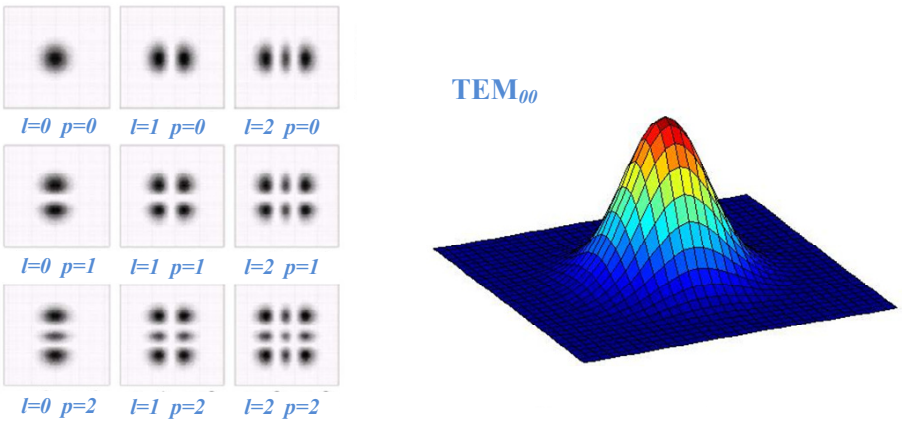


Abbildung 2.5: TEM_{lp} Moden in 2D, Gaußsches Energieprofil einer TEM_{00} Mode.

Laser-Material Interaktion Wie konventionelles Licht, wird ein Laserstrahl beim Auftreffen auf ein Material, abhängig von der Wellenlänge und Intensität des Lasers sowie von einigen Materialparametern (u.a. der Plasmafrequenz bei Metallen sowie dem Absorptionskoeffizienten) reflektiert, transmittiert oder absorbiert. Letzteres hat die größte Bedeutung in der Lasermaterialbearbeitung. Bei der Absorption wird der Laserstrahl bis zu einer gewissen Eindringtiefe vom Material absorbiert, Atome und Moleküle des Materials werden zu Schwingungen angeregt wobei Wärme erzeugt wird. Die Intensität der eingebrachten Energie nimmt exponentiell mit dem Inversen des Absorptionskoeffizienten ab, so dass bei Stahl die Eindringtiefe beispielsweise nur einige Nanometer beträgt. Die erzeugte Wärme wird, abhängig von der Wärmeleitfähigkeit des Materials, weitergeleitet und kann zu Phasenübergängen führen, das Material schmilzt oder verdampft. Der entstehende Dampf kann mit dem Laserstrahl interagieren, wobei eine Laser-Plasma Interaktion auftreten kann. Nach Vorbild dieses Prozesses werden, vor allem im industriellen Umfeld, Metalle geschnitten und geschweißt. Meist werden hierfür Dauerstrahllaser verwendet.

Laser-Knochen Interaktion Der Prozess des Metallschneidens mit Laser ist aufgrund der erhöhten Heterogenität biologischen Gewebes nur teilweise auf dieses übertragbar. Absorption in biologischem Gewebe hängt vor allem von den Parametern Energie, Expositionszeit, Energiedichte (Fokusdurchmesser) und Wellenlänge ab, wobei letzteres das entscheidende Kriterium darstellt. Knochen absorbiert beispielsweise wegen seiner hohen Wasser- und

Hydroxylapatitanteile infrarote Strahlung (1 um-1 mm) und kann daher besonders gut mir Lasern aus diesem Bereich wie Er:YAG (2.9 μm) oder CO_2 (10, 6 μm) bearbeitet werden. Abhängig von Expositionszeit und Energiedichte treten während der Absorption mechanische, thermische und photochemische Prozesse auf, die wiederum verschiedene Gewebereaktionen wie Plasmabildung, explosionsartiges Abtragen, Verdampfung, Koagulation und photodynamische Prozesse nach sich ziehen (vgl. Abbildung 2.6).

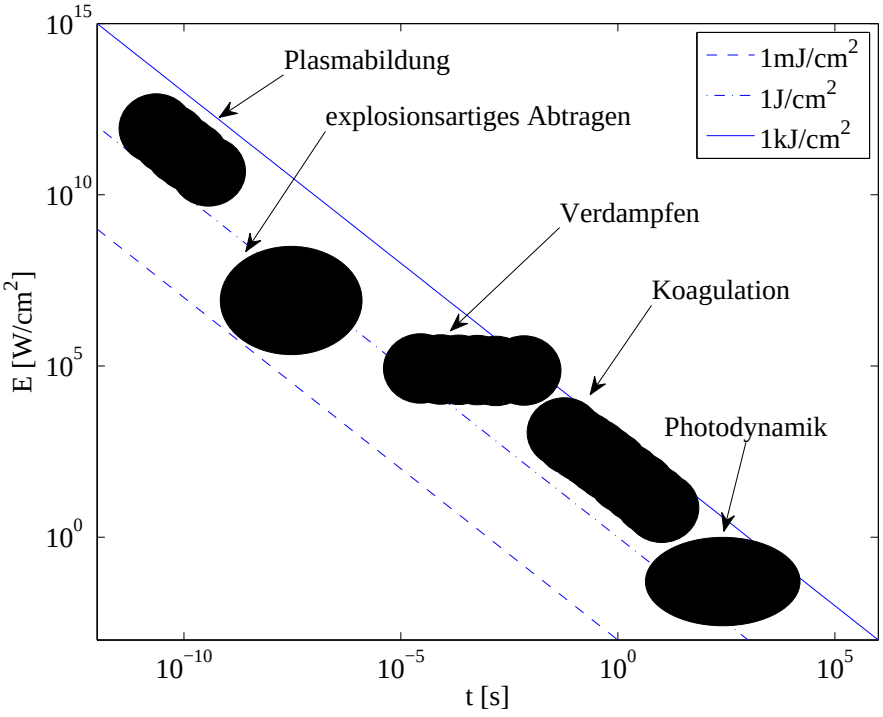


Abbildung 2.6: Hartgewebe - Laser Wechselwirkungsmechanismen abhängig von der Bestrahlungsstärke $E\ [W/cm^2]$ und der Bestrahlungszeit $t\ [s]$.

Zwar wurden bereits 1964 von Lithwick et al. [36] erste Experimente zur Wechselwirkung von Knochen und Laser gemacht, doch galt das Knochenschneiden mit Laser lange Zeit als nicht realisierbar. Erst Engelhart et al. [16] widmeten dem Thema Laserosteotomie eine ganze Veröffentlichung und beschrieben darin unter anderem Versuche bei denen ein 10-35 mm dicker Knochen mit einem 250 W CO_2 Laser mit einer Überfahrung durchgeschnitten wurde. Die dabei auftretenden Aufschmelzungen wurden zu damaliger Zeit als unkritisch betrachtet. Erst Verschueren [62] beschrieb kurz darauf Probleme, welche beim Heilungsprozess durch solche Aufschmelzungen und thermischen Schädigungen bei der Laserosteotomie auftraten. Charlton zeigte Anfang der 90er Jahre [11], dass dieses Phänomen zwar auch bei der Verwendung von Er:YAG Lasern auftritt, jedoch in geringerem Maße als bei CO_2 Lasern, da bei ersterem ein günstigeres Verhältnis zwischen thermischer Relaxationszeit und der Zeit bis zum Beginn des Knochenabtrags besteht [5]. Jedoch traten auch in diesem Fall nekrotische Grenzschichten von mehreren μm auf. Da die thermische Schädigung stark abhängig davon ist, in welcher Zeit die Energie in den Knochen gebracht wird, versucht man diesen Effekt heute mit möglichst kurzen und rechteckigen Pulsen zu vermeiden, welche die Energie so schnell in das Material einbringen, dass kaum thermische Diffusion auftreten kann. Um die thermischen Effekte weiter

zu verringern, wird die zu bearbeitende Stelle mit einem Wasserstrahl gekühlt. Dabei kommt es zu Absprengungen mikroskopisch kleiner Knochenteile indem das im Knochen enthaltene Wasser durch die Lasereinwirkung explosionsartig verdampft, was u.a. Afilal am Beispiel des CO_2 Lasers als thermo-mechanisches Abtragsmodell beschreibt [1].

2.1.3 Einsatz von Robotik

Robotik in der industriellen Laserbearbeitung Roboter übernehmen im industriellen Umfeld heutzutage vielfältige Aufgaben. Vom Widerstandsschweißen, das schon in den sechziger Jahren erstmals mit einem Industrieroboter durchgeführt wurde, bis hin zur Montage und zur Handhabung sind Roboter nicht mehr aus den Produktionsstätten wegzudenken. In Deutschland kommen heute ca. 140.000 Industrieroboter zum Einsatz [28]. Zu nennende Vorteile sind dabei die Erhöhung der Produktivität, verbesserte Wiederholgenauigkeit und Präzision, sowie die Verbesserung der Arbeitsbedingungen, indem monotone oder gesundheitsschädigende Arbeiten automatisiert werden. Von den vielfältigen Möglichkeiten der Materialbearbeitung mit Laser werden hauptsächlich Laserschweißen und Laserschneiden in der Industrie mit Robotern durchgeführt. Ein aktueller Forschungsschwerpunkt dabei ist das Remote Laserstrahlschneiden (RLC: Remote Laser Cutting) und Remote Laserstrahlschweißen (RLW: Remote Laser Welding). Während beim herkömmlichen Laserstrahlschneiden bzw. -schweißen der durch kurze Brennweiten konventioneller Laserstrahlenquellen bedingte geringe Arbeitsabstand die restriktive Randquelle war, ermöglichen neue Faserlaser mit Arbeitsabständen bis zu zwei Metern beim Schweißen und $40\ cm$ beim Schneiden (vgl. Abbildung 2.7) höhere Arbeitsgeschwindigkeiten, mehr Prozessflexibilität und geringere Kollisionsrisiken [66].

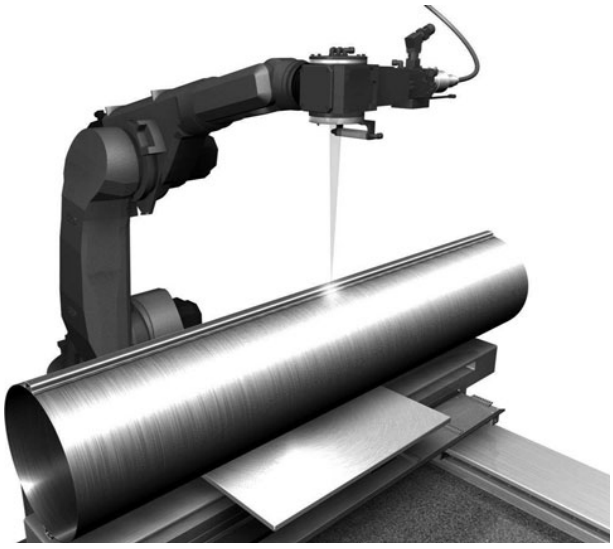


Abbildung 2.7: Remote Laserstrahlschweißen, entnommen aus [24].

Eine der dabei auftretenden Schwierigkeiten ist die erhöhte Anforderung an die Roboterkinematik. Bei solch großen Abständen können bereits kleinste Bewegungen des Laserendeffektors große Spurfehler des Schnitts bewirken, dessen Durchmesser teilweise unter $100\ \mu m$ liegt [67]. Da beim Schneiden mehrmals über dieselbe Stelle gefahren werden muss ist eine

hohe Wiederholgenauigkeit nötig. Auch Methoden zur Trajektorien- und Aufgabenplanung solcher Systeme unterscheiden sich wesentlich von denen konventionellen robotergeführten Laserstrahlschneidens und –schweißens. Hohe Robustheit bezüglich Winkel- und Fokusabweichungen beim RLW bedeuten Freiheitsgrade für die Planung, während die erhöhten Anforderungen an die Positioniergenauigkeit beim RLC Restriktionen mit sich bringen. Weder konventionelle Onlineprogrammiermethoden (manuelles Einprogrammieren der Bahn indem einzelne Punkte per Handeingabegerät angefahren werden) noch Offlineprogrammiermethoden (Programmierung der Bahn in einer virtuellen Umgebung) genügen diesen Anforderungen. Stattdessen können Methoden eingesetzt werden, bei denen der Bediener die Schweißnähte oder Schnittlinien nur noch als abstrakte Aufgabe definiert. Die entsprechenden Trajektorien werden dann über komplexe Planungsalgorithmen roboterspezifisch optimiert [24], [45]. Neben einer höheren Schweißnaht- und Schnittqualität wird mithilfe solcher Planungsalgorithmen die Kombination von RLC und RLW mit demselben Roboter-Laser-System möglich, was eine weitere Erhöhung der Bearbeitungstaktzeit ermöglicht.

Robotik in der Chirurgie Anders als bei industriellen Anwendungen ist in der chirurgischen Robotik nicht die Automatisierung von Arbeitsvorgängen das Hauptziel. Anstatt einprogrammierte Handlungen vollautomatisch auszuführen, wird der Roboter im klinischen Umfeld als chirurgisches Werkzeug betrachtet, welches meist aktiv bedient wird und immer unter direkter Kontrolle des Menschen steht. Ein Chirurgieroboter kann dem Arzt beispielsweise ermöglichen, genauer und präziser, zeitsparender oder minimalinvasiver zu arbeiten [59]. Forschung und Entwicklung laufen in diesem Bereich meist applikationsspezifisch und in interdisziplinärer Zusammenarbeit zwischen Ärzten und Ingenieuren ab. Die Anforderungen auf Ingenieursseite sind hoch, nur bei messbaren und signifikanten Vorteilen eines Systems besteht die Chance der Akzeptanz auf Seiten der Mediziner. Außerdem müssen für die Marktzulassung von Medizinrobotern wie Medizinprodukten im allgemeinen sehr strenge Sicherheitsstandards erfüllt werden [49], weshalb die Entwicklungszeiten meist länger sind als bei Industrierobotern. Die Geschichte der medizinischen Robotik begann mit einer neurologischen Anwendung im Jahr 1988, als Kwoh [33] mithilfe eines PUMA Industrieroboters eine Biopsienadel im Gehirn setzte. Daraus entstand der erste kommerzielle chirurgische Roboter Neuromate, welcher 1999 eine FDA (U.S. Food and Drug Administration) Zulassung bekam und auch heute noch für verschiedene neurologische Eingriffe verwendet wird [20]. Der Neuromate wird zu jederzeit direkt vom Chirurgen gesteuert. Bald darauf wurde das System RoboDOC entwickelt, mit dem Fräsungen im Femur für das Setzen von Hüftgelenksendoprotesen bzw. im Knie für das Setzen von Knieendoprothesen durchgeführt werden konnten. Der Roboter arbeitete dabei teilautonom. Die damit erreichbaren Resultate waren allerdings nicht signifikant besser und mit technischen Komplikationen verbunden, weshalb es zu einer Welle der Kritik kam und das System in Europa und Amerika kaum mehr eingesetzt wird [50]. Wegen dem Misserfolg des RoboDOC–Systems ging fortan der Trend wieder weg von Systemen, welche aktiv und autonom arbeiten und hin zu passiven Systemen. Stattdessen basieren heute eingesetzte chirurgische Roboter meist auf dem Prinzip der Telemanipulation. Dabei wird die Bewegung, die der Chirurg an einer Konsole ausführt, an den Roboter weitergeleitet, Chirurg und Roboter sind dabei durch eine physikalische Barriere getrennt. Die medizinische Anwendung des Prinzips der Telemanipulation oder Telepräsenz, welches auch im Bereich der Weltraumforschung oder Reaktortechnik eine Rolle spielt, war ursprünglich dadurch motiviert, medizinische Eingriffe an schwer zugänglichen

Orten wie Kriegsschauplätzen, Naturkatastrophen oder dem Weltall ohne Anwesenheit eines Arztes zu ermöglichen [3], [23]. Bald wurden Möglichkeiten in der minimalinvasiven Chirurgie deutlich [26], wo die physikalische Barriere nicht weite Entfernungen sind, sondern die Körperhülle. Bei der minimalinvasiven Chirurgie, auch als Knopflochchirurgie bezeichnet, werden Instrumente und Optik durch kleine Hauteinschnitte in den Körper gebracht um dort zu operieren. Dadurch entstehen kleinere Wunden als bei der offenen Chirurgie und der Heilungsprozess wird deutlich beschleunigt. Für den Chirurgen treten dabei Schwierigkeiten auf, wie die eingeschränkte Haptik. Er hat keinen direkten Zugang zum Operationssitus und kann die Instrumente nur über den Einstichpunkt bewegen, wodurch sich die Bewegungen umkehren, Freiheitsgrade eingeschränkt sind und nur ein kleiner Bereich zugänglich ist. Über einen der Hauteinschnitte wird ein Endoskop eingeführt, welches oft von einer anderen Person geführt wird und dessen Bild der Chirurg auf einem Monitor sehen kann. Dieses Auseinanderliegen von Optik und Haptik wirkt weiter erschwerend für den Chirurgen. Mithilfe von Telepräsenz und Robotik sind diese Schwierigkeiten zum Teil überwindbar. Abbildung 2.8 zeigt den Roboter DaVinci der Firma Intuitive Surgical, welcher u.a. für urologische Eingriffe verwendet wird.

Abbildung 2.8: Der Operationsroboter DaVinci von Intuitive Surgical.

Wie auf der Abbildung erkennbar, operiert der Chirurg an einer Konsole. An dieser wird ihm die über das Endoskop aufgenommene Optik in 3D auf einem Monitor dargestellt, über zwei Joysticks manipuliert er die Instrumente, welche sich im Körper befinden. Es ist eine Kraftrückkopplung der Manipulation möglich, d.h. am Instrument auftretende Kräfte sind am Joystick spürbar (Force-Feedback). Außer die Überwindung der eben beschriebenen Schwierigkeiten in der minimalinvasiven Chirurgie ergeben sich daraus weitere Möglichkeiten. Über Augmentierung können dem Chirurgen weitere Informationen wie präoperativ aufgenommene Computertomographiedaten dargestellt werden. Darüber hinaus kann zwischen der vom Chirurgen vorgegebenen und der am Instrument ausgeführten Bewegung skaliert werden, wodurch ein genaueres Arbeiten möglich wird. Aufgrund seiner Größe, seinem Gewicht sowie dem wenig modularen Aufbau ist das DaVinci System allerdings sehr unflexibel und die Integration in bestehende Operationsabläufe bedeutet großen Aufwand, weshalb der Roboter an einem Standort meist nur für eine Operation genutzt wird und damit ein sehr spezialisiertes System darstellt. Im Projekt MiroSurge (vgl. Abbildung 2.9) wird versucht diese Nachteile zu überwinden [21]. Die Hauptbestandteile des dabei entwickelten Systems sind eine Eingabekonsole und, je nach

Anwendung, ein bis drei MIRO Leichtbauroboter. Mit einem Gewicht von ca. $10\ kg$ sind die Roboter leicht in ihrer Position zu verändern. Das System soll für verschiedene minimalinvasive und offene Operationen wie die Laparoskopie, die Laserosteotomie und das Setzen von Biopsienadeln im Gehirn einsetzbar sein. Im Gegensatz zu spezialisierten Systemen wie DaVinci oder RoboDOC soll es somit ein universelles und flexibles Werkzeug im Operationssaal darstellen [22].

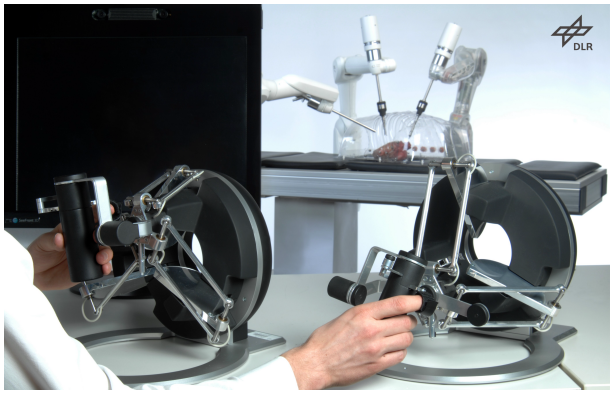


Abbildung 2.9: Das System MiroSurge, entwickelt am Deutschen Zentrum für Luft und Raumfahrt.

Neben den schon erwähnten Vorteilen eines Medizinroboters, welcher mit Telepräsenz arbeitet, wie Skalierung und Augmentierung, werden bei MiroSurge an weiteren Möglichkeiten, den Arzt zu unterstützen, geforscht. Ein Beispiel ist die Kompensation der Herzbewegung bei kardiologischen Operationen am schlagenden Herzen [43], dessen Bewegungen aufgenommen und herausgefiltert werden. So kann der Chirurg über die Konsole an einem quasi stillstehenden Herzen operieren. Ein weiteres Beispiel ist die Untersuchung verschiedener unterstützender Anzeigemodi für den Chirurgen, wie das Übermitteln der am Instrument auftretenden Kräfte über haptisches und optisches Feedback [60]. Trotz der kurzen Geschichte der Verwendung von Medizinrobotern zeigt der kommerzielle Erfolg der wenigen auf dem Markt verfügbaren Systeme [12] einen stark ansteigenden Trend, welcher im Zuge des steigenden Bedarfs kostensparender und effizienterer Methoden im Gesundheitswesen eine ähnliche Entwicklung nehmen könnte wie die computergestützte Produktion der Industrie vor 20 Jahren [58]. Um diese Nachfrage bewältigen zu können müssen die Systeme allerdings vorerst kleiner und bedienerfreundlicher werden um einfacher in den Operationsraum integriert zu werden, sowie eine höhere Flexibilität bezüglich ihrer Einsatzmöglichkeiten haben [21], [52].

Robotik in der Laserosteotomie Einige Arbeiten zur robotergestützten Laserosteotomie, wie die Experimente von Wörn et al. [64], wurden am Institut für Prozessrechentechnik, Automation und Robotik der Universität Karlsruhe durchgeführt. Wörn führte mithilfe eines Stäubli RX90B eine Laserosteotomie mittels scankopfgeführtem CO_2 Lasers an Schweinekadavern durch. Dabei wurde allerdings weder ein computergestütztes Planungssystem entworfen, noch wurden die Lage des Versuchstieres und des Roboters zueinander registriert, sondern es wurde nur die Möglichkeit einer robotischen Laserosteotomie gezeigt. Kahrs zeigte in seiner Dissertation [29] eine robotergestützte Cochleaosteotomie (Öffnung des Innenohrs). Der Prozess, bei dem ein

zylindrischer mikrochirurgischer Kanal erzeugt wird, ist eher als Laserbohren statt als Laserschneiden zu betrachten. Der Knochen wird dabei Schicht für Schicht mit einem gepulsten, scankopfgeführten CO_2 Laser abgetragen, wobei dies zuerst mittels eines selbstentwickelten Planungssystem und Laserabtragsmodells geplant und simuliert wird. Es wurde eine bildgestützte Regelung entworfen, welche die Tiefenentwicklung überwacht und mit dem Modell abgleicht. Eine Registrierung zwischen Knochen und Roboter wird nicht beschrieben. Der Fokus der Arbeit liegt darauf, die Tiefenentwicklung zu überwachen, um nicht auf eine unter dem Knochen liegende Membran zu treffen, und dabei eine optimale Verteilung der Einzellaserpulse zu erreichen. Ein vollständiges System mit mathematischem Lasermodell, Planungsmöglichkeit und Registrierung wird von Burgner [7] beschrieben (vgl. Abbildung 2.10). Dabei wird allerdings keine intraoperative Regelungsmöglichkeit vorgestellt, sondern nur eine Möglichkeit gezeigt, diese zu integrieren. Die Planung kann anhand von CT-Daten (Computertomographie) durchgeführt werden. Mit dem mathematischen Lasermodell können die Positionierung der Pulse und damit die vom Roboter anzufahrenden Scankopfpositionen optimiert werden (auch hier wird ein gepulster, scankopfgeführter CO_2 Laser verwendet). Das System wurde unabhängig vom Roboter entwickelt, Experimente wurden mit einem Stäubli RX90B CR sowie mit einem KUKA LWR an Tierkadavern durchgeführt.

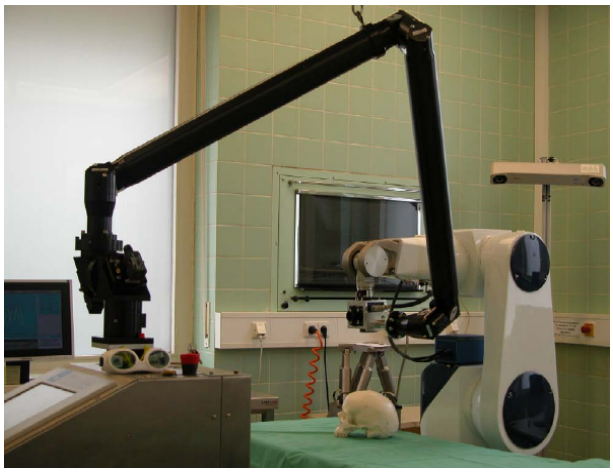


Abbildung 2.10: Robotergestütztes Laserosteotomiesystem mit einem Stäubli RX90B CR Roboter, entnommen aus [7].

Aufbauend auf dieses System (KUKA LWR, gepulster, scankopfgeführter CO_2 Laser) stellen Mönich und Stein eine auf optischem Tracking des Roboters und präoperativ gewonnenen CT-Daten basierte Positionsregelung [41] vor, sowie eine Möglichkeit zur schnellen und einfachen Registrierung per Handführung des Roboters [55]. Neben den beschriebenen Ansätzen der robotergestützten Laserosteotomie befassen sich einige Arbeiten damit, die Nachteile der handgeführten Laserosteotomie (fehlende Haptik und fehlende Tiefenentwicklungskontrolle, keine Kontrolle über Position und Pose des Lasers zum Knochen) durch Sensorik und modellbasierte Regelungskonzepte zu vermindern. Rupprecht [46] verwendet piezoelektrische Sensorik auf der Knochenoberfläche zur Detektion von Vibrationen sowie eine Photodiode zur Detektion von Lichtphänomenen, welche charakteristisch sind beim Übergang eines Er:YAG Lasers von Kompakta zu Spongiosa. Bei Erreichen dieses Überganges wird der Laser abgeschaltet. Stopp stellt in seiner Dissertation [56] ein Konzept für einen durch Navigation geregelten

manuellen Hartgewebeabtrag mit einem Er:YAG Laser in der dentalen Implantologie vor. Der manuell geführte Laserendeffektor wird dabei bezüglich Position und Pose optisch getrackt (vgl. Abbildung 2.11), sobald der Laserstrahl außerhalb der geplanten Kavität liegt, wird der Laser ausgeschaltet. Präoperativ wird in einer Planungsumgebung das Abtragsvolumen anhand von CT-Daten festgelegt, die anatomischen Daten werden vor der Operation mit der Realität registriert. Parallel zur Operation wird in einer Simulation der Abtrag mit einem mathematischen Modell berechnet und dem Arzt visualisiert. Konzepte für eine Sensorintegration um die Tiefenentwicklung zu berücksichtigen und das Modell intraoperativ anzupassen werden nicht vorgestellt.

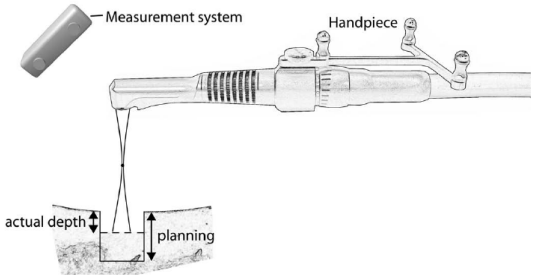


Abbildung 2.11: Intraoperative Laserabtragsregelung durch Navigation von Stopp, entnommen aus [56].

Das System ist auf das Abtragen eines Implantatkanals im Kiefer spezialisiert und nicht direkt auf das Lasern von Schnitten übertragbar, da das manuelle Führen des Laserendeffektors den geforderten Genauigkeiten eines Laserschnitts nicht genügt.

2.2 Planung für die robotergestützte Laserosteotomie

In Unterkapitel 2.1 wurden bereits einige Arbeiten zur Laserosteotomie zusammengefasst, wobei erkenntlich wurde, dass alle Arbeiten, welche ein vollständiges System zur Laserosteotomie vorstellen [7] [29] [56], eine Planungsmethode beinhalten. Dies resultiert aus einigen Eigenschaften des Laserschneideprozesses. Zum einen ermöglicht dieser, anders als der Schnitt mit einer Säge, keine Haptik, ohne welche die Abschätzung der Tiefe schwierig bis unmöglich ist. Zum anderen sind die Vorteile der präzisen Schnittführung letztendlich nur mit einer sehr genauen und ruhigen Führung des Laserendeffektors durch einen Roboter zu erzielen, was präoperativ geplant werden muss. Um die Planung für eine robotergestützte Laserosteotomie in die bestehenden präoperativen Planungsmethoden zu integrieren, werden zusätzliche Schritte benötigt, wie das Erfassen der anatomischen Strukturen durch dreidimensionale Bildgebung, sowie die Modellierung dieser als Abtragsvolumen. Zur Planung und Berechnung des Abtrags muss außerdem der Hartgewebeabtrag durch den Laser modelliert werden. Damit der Roboter den geplanten Schnitt durchführen kann, muss die Trajektorie mathematisch beschrieben werden. Im Folgenden sollen zuerst bestehende Planungsmethoden zur Osteotomie erfasst und notwendige Änderungen zur Integration der Laserosteotomie diskutiert werden. Im Anschluss werden einige Forschungsarbeiten aus dem Bereich der Planungsmethoden für die robotergestützte Laserosteotomie zusammengefasst.

2.2.1 Planungsmethoden Osteotomie

Wie in anderen klinischen Feldern richten sich die Planungsmethoden in der Osteotomie stark nach der Anwendung und lassen sich nicht verallgemeinern. Beispiele für Anwendungen sind die Knie- und Hüftgelenksendoprothetik, Umstellungsosteotomien bei Hüftgelenksfehlstellung, sowie kieferorthopädische Umstellungsosteotomien wie die Le-Fort Osteotomie. Je nach klinischem Zentrum und Komplexität der Operation sind dabei zwei Hauptkategorien der Planung zu erkennen: Konventionelle Planungsmethoden und computergestützte Planungsmethoden, welche z.B. notwendig werden, wenn die Operation mithilfe von Navigation oder Robotik durchgeführt wird. Bei der Navigation werden Position und Pose von Instrument und Patient z.B. optisch erfasst und auf einem Bildschirm dargestellt. Damit sind genauere Eingriffe möglich. Konventionelle Planungsmethoden kommen meist ohne anatomische 3D Daten aus. Bei der Knieendoprothetik können z.B. 2D Röntgenaufnahmen angefertigt und die Änderung und geplante Positionierung des Implantats in diese eingezeichnet werden (vgl. Abbildung 2.12). Die Skalierung der Röntgenaufnahmen ist standardisiert, so dass vom Implantathersteller bereitgestellte Schablonen verwendet werden können. Auch intraoperativ wird mit Schablonen und Messlehren gearbeitet [63].

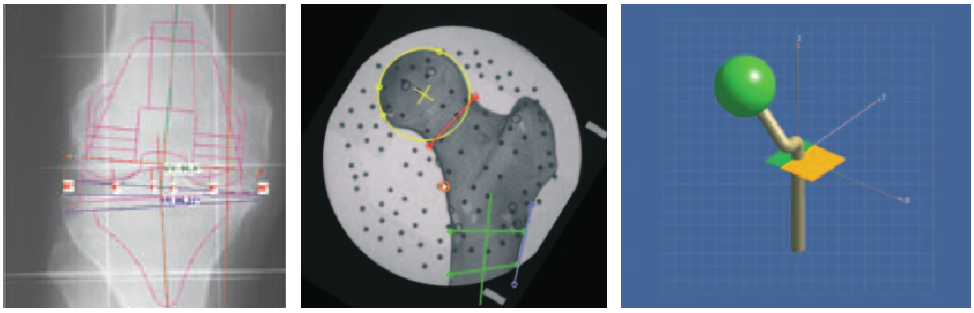


Abbildung 2.12: Planungsmethoden mit 2D Röntgenaufnahmen. Links: Unter Verwendung von Implantatsschablonen, entnommen aus [63]. Mitte und rechts: Generierung eines vereinfachten 3D Modells aus 2D Aufnahmen, entnommen aus [18].

Eine Möglichkeit zur computergestützten Umstellungsosteotomie am Femur mittels 2D Daten wird von Gottschling und Burgkart [18] beschrieben. Dafür muss intraoperativ ein optischer Tracker am Femur befestigt werden. Mithilfe eines C-Bogens wird dann eine Saggital- sowie eine Frontalröntgenaufnahme (anatomische Ebenen, saggital: von der Seite, frontal: von vorne) des Femurs erzeugt und daraus ein vereinfachtes 3D Model generiert (vgl. Abbildung 2.12). An diesem kann die Umstellung geplant und sich ergebende Schnittebenen können automatisch errechnet werden. Darauffolgend wird mit einem navigierten Schneidwerkzeug der Schnitt durchgeführt, eine Referenzierung ist nicht notwendig, da der am Femur angebrachte Tracker mitgeröntgt wurde. Ömürlö [68] beschreibt eine computergestützte Planungsmethode für die Hüftgelenksendoprothetik, bei der ebenfalls nur jeweils eine 2D Röntgenaufnahmen der Saggital- sowie der Frontalebene verwendet werden um die optimale Lage des künstlichen Hüftgelenks mithilfe der Planungssoftware medi-CAD von HecTec zu ermitteln. Mit der Software können Implantate verschiedener Hersteller eingepasst werden, außerdem kann die Position bezüglich ihrer biomechanischen Eigenschaften optimiert und die postoperative Beinlänge errechnet werden. Die Verwendung von 3D Daten wird z.B. beim Einsatz des Roboters RoboDOC für

die Hüftgelenksendoprothetik verwendet [9]. Die Planungsmethodik unterscheidet sich vorerst nicht sehr von der Methodik Ömürlös. Auch hier werden in einer Software (Orthodoc) vorerst nur Schnittbilder des 3D Datensatzes verwendet um die gewünschte Positionierung des künstlichen Hüftgelenks mit digitalen Schablonen des Hüftgelenksherstellers festzulegen. Die Position kann von der Software mit einer biometrischen Analyse optimiert werden. Da die eigentliche Operation allerdings von einem Roboter ausgeführt wird, müssen die anatomischen Daten in 3D vorliegen um die Planung intraoperativ mit der Realität registrieren zu können. Dies geschieht indem der Roboter den Knochen zuerst durch Anfahren mehrerer Punkte abtastet und so seine Lage und Orientierung erkennen kann. Methoden zur computergestützten Navigation in der Orthopädie ohne Roboter werden z.B. von Shi und Lüth vorgestellt [51]. Die 3D Daten werden wie beim RoboDOC System dazu verwendet, um eine präoperative Planung mit der intraoperativen Realität zu registrieren. Die eigentliche Planung geschieht hierbei meist anhand von 2D Schnittebenen durch das 3D Volumen. Der Vorteil der 3D Daten ist jedoch, dass diese Schnitte beliebig im Volumen orientiert und positioniert werden können und nach abgeschlossener Planung ein simuliertes postoperatives Ergebnis in 3D beurteilt werden kann.

2.2.2 Planungsmethoden Laserosteotomie

Die erwähnten konventionellen sowie computergestützten Planungsmethoden befassen sich fast ausschließlich mit der Positionierung von Implantaten und Instrumenten relativ zur bestehenden anatomischen Struktur. Dazu sind geometrische Informationen relevant, jedoch muss das Knochenmaterial nicht beschrieben oder modelliert werden, da die Änderung dessen z.B. durch Interaktion mit einem Instrument nicht beschrieben oder simuliert wird. Zwar wird beispielsweise beim RoboDOC der Fräsprozess zum Fräsen der Implantatshöhle geplant, dabei wird allerdings nur die Lage des Fräsinstrumentes zum Knochen sowie am Fräsinstrument auftretende Kräfte berücksichtigt und nicht der Knochenabtrag an sich berechnet oder simuliert.
Zur Planung einer Laserosteotomie reicht eine rein geometrische Beschreibung der anatomischen Struktur nicht aus. Beim Hartgewebeabtrag durch Laser besteht weder mechanischer Kontakt zwischen Laserwerkzeug und Gewebe, noch ermöglicht haptisches oder mechanisches Feedback Aufschluss über die momentane Schnitttiefe. Deshalb muss der Schnittprozess modelliert und simuliert werden. Dazu ist es notwendig, das Knochengewebe an jeder Stelle im Operationsbereich bezüglich seiner Materialparameter zu modellieren. Dies ist am besten mit einem Voxelmodell möglich, bei dem gemäß einer bestimmten Diskretisierung jedem Punkt im Knochenvolumen Eigenschaften wie Dichte und momentaner Abtrag durch den Laser zugeordnet werden. In der in Abschnitt 2.1 beschriebenen Arbeit von Kahrs wird das umgesetzt [29]. Nach Festlegung des Abtragsbereichs, welcher in diesem Fall einen zylindrischen Kanal darstellt, wird dieser in einzelne Voxel unterteilt. Verschiedene Knochendichten bzw. der Übergang von kompaktem zu spongiösem Material werden hierbei nicht berücksichtigt. Der Gesamtabtrag wird als die Summe der vom gepulsten CO_2 - Laser abgegebenen Einzelpulse beschrieben. Ziel der Planung ist es die Einzelpulse optimal zu verteilen. Jeder Einzelpuls wird als ein Energieeintrag in das Voxelvolumen betrachtet. Es wird davon ausgegangen, dass der Laserstrahl senkrecht einfällt und der Fokus des Laserstrahls auf dem Material liegt. Der Vorgang erfolgt schichtweise, die Schichtdicke richtet sich nach der maximalen Tiefe eines Laserpulses. Die Verteilung der Einzelpulse reduziert sich so auf ein zweidimensionales Packproblem.

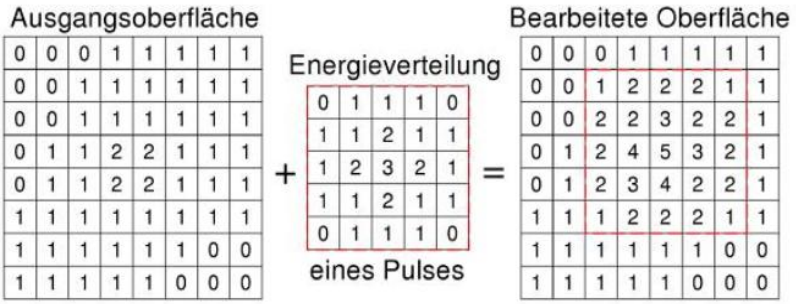


Abbildung 2.13: Energieeintrag eines Einzelimpulses auf eine Voxelschicht nach Kahrs [29].

Ein anderer Ansatz findet sich bei Stopp [56]. Wie in Abschnitt 2.1.3 beschrieben wird hierbei der Abtrag eines für die Zahnimplantation gedachten Kanals im Kiefer durch einen handgeführten Er:YAG Laser beschrieben. Das Laserabtragsmodell ist in seiner Anwendung flexibler als Kahrs Modell. Auch er modelliert den Gesamtabtrag als die Summe von Einzelpulsabträgen. Der Einfallswinkel des Laserstrahls wird bei Stopp nicht berücksichtigt, ebenso wenig wie die Entstehung von Debris. Als Debris werden die beim Abtrag abgesprengten Trümmer bezeichnet, welche einen Teil der Abtragsstelle bedecken und so den Prozess behindern können. Jeder Gesamtenergieeintrag wird als gaußscher Energieeintrag beschrieben, so dass, abhängig vom radialen Abstand und der Positionierung entlang der optischen Achse jedes Voxels, dessen Energieeintrag angegeben werden kann. Das Abtragsvolumen wird als Voxelvolumen beschrieben, wobei jedes Voxel eine feste Größe ($0,22 \times 0,22 \times 0,33\ mm$) und einen bestimmten Gesundheitszustand zugewiesen bekommt. Der Gesundheitszustand verringert sich je nach eingebrachter Energie. Sobald er Null ist, gilt das Voxel als entfernt. So werden bei jedem Einzelimpuls die getroffenen Voxel ermittelt und entsprechend des Modells abgetragen (vgl. Abbildung 2.14). Aufgrund begrenzter Rechenleistung wird nicht das ganze Knochenvolumen als Voxel modelliert, sondern nur die geplanten Kavitäten, welche ca. 30.000 Voxel beinhalten können.

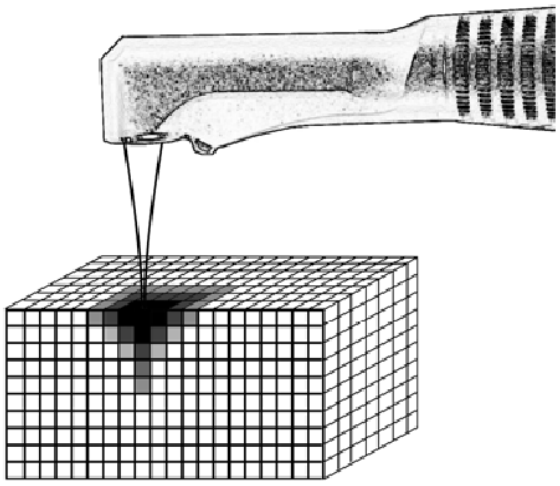


Abbildung 2.14: Energieeintrag in ein 3D Voxelmodell nach Stopp [56].

Die Planungsmethoden von Kahrs und Stopp beziehen sich auf das Abtragen eines größeren Volumens, nicht auf das Lasern von Schnitten. Das Lasern von Schnitten wird wie in Abschnitt 3.1.3 beschrieben von Burgner realisiert [7]. Die Planung und das entsprechende Modell unterscheiden sich sehr von denen Stopps und Kahrs. Burgner arbeitet nicht mit Voxelmodellen sondern mit Oberflächenmodellen. Sie modelliert den Schnitt als Summe mehrerer Überfahrungen. Eine Überfahrung besteht aus einer Aneinanderreihung von Einzelpulsen, welche eine konstante Überlappung haben, sodass eine konstante Tiefenentwicklung entlang der Überfahrung angenommen werden kann. Als erster Schritt der Planung werden an einem 3D Oberflächenmodell Schnitttrajektorien definiert, wie in Abbildung 2.15 zu sehen ist. Entlang dieser Trajektorien wird jedem Punkt eine gewisse Tiefe zugeordnet. So ergibt sich über das Abtragsmodell die notwendige Anzahl von Einzelimpulsen pro Punkt und damit die notwendige Anzahl von Überfahrungen pro Streckenabschnitt. Während der Abtrag des Volumens bei Kahrs Schicht für Schicht und bei Stopp nach jedem Einzelpulsabtrag generiert wird, berechnet Burgner prädiktiv die nötigen Einzelpulsabträge pro Punkt der interpolierten Trajektorie, um die dazu nötigen Überfahrungen durch den Roboter zu ermitteln. Es wird ein Lasermodell verwendet, welches die Tiefenentwicklung an einem Punkt als Funktion der dort aufgebrachten Anzahl von Einzelpulsabträgen betrachtet. Dabei entsteht eine logarithmische Tiefenentwicklung, bei der die Entstehung von Debris, der wachsende Energieverlust an den Kraterwänden sowie einige andere Phänomene berücksichtigt werden. Sie legt Grenzen für den zulässigen maximalen Lasereintrittswinkel sowie die Abweichung vom Fokus fest, die bei der Überfahrung durch den Roboter eingehalten werden müssen.

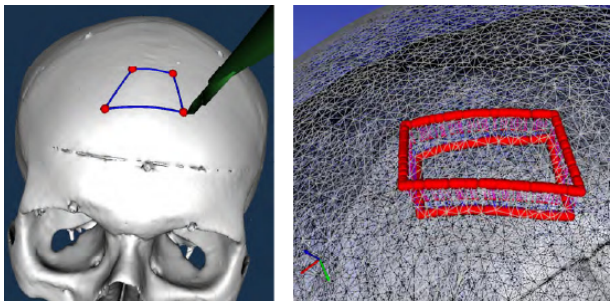


Abbildung 2.15: Planen einer Schnitttrajektorie nach Burgner [7]. Dabei wird jedem Punkt auf der Trajektorie eine Tiefe zugeordnet.

Kapitel 3

Motivation

Im Folgenden werden die Vor- und Nachteile der Laserosteotomie zusammengefasst und der in Kapitel 2 beschriebene Stand der Technik kritisch beurteilt. Daraus werden Handlungsschritte abgeleitet und Ziele formuliert, mit welchen diese Arbeit zur Verbesserung der robotergestützten Laserosteotomie beitragen soll. Das Knochenschneiden mit Laser beinhaltet folgende Vorteile:

- Höhere Genauigkeit und exaktes Schneiden gemäß präoperativ geplanter Daten, daher kann z.B. ein Implantat vorab gefertigt werden
- Geringere Schnittbreite und damit geringerer Verlust an gesundem Knochenmaterial
- Beliebige patienten- und fallspezifische Schnittformen, u.a. erzielen geschwungene Strukturen höhere Stabilität
- Geringeres Trauma und Senkung des Infektionsrisikos, da kein physischer Kontakt von Instrument und Patient stattfindet

Aufgrund folgender unüberwundener Nachteile hat sich die Laserosteotomie bisher aber nicht als praxistauglich erweisen können.

- Langsame Schnittgeschwindigkeit
- Entstehung von Nekrosen durch hohen thermischen Energieeintrag in das Knochengewebe. Diese ist u.a. durch Abweichung vom Fokusabstand sowie der Abweichung des Laserstrahls von der Oberflächennormale bedingt
- Fehlende Haptik und daher keine Kontrolle über die Tiefenentwicklung
- Begrenzte Realisierbarkeit der geplanten, exakten Schnittgeometrie durch manuelles Führen des Lasers

Die in Kapitel 2 vorgestellten Ansätze behandeln die Überwindung dieser Nachteile nur zum Teil. Burgner [7] stellt ein sehr umfassendes System vor, welches eine Planung des Schnitts, die Registrierung von Planung und Realität sowie den Einsatz eines Roboters zur Durchführung

komplexer, geschwungener Schnittgeometrien beinhaltet. Jedoch ist das von ihr vorgeschlagene Abtragsmodell oberflächenbasiert, weshalb der Abtrag intraoperativ nicht visualisiert werden kann und die Knochendichte sowie mögliche Hohlräume nicht modelliert werden können. Das Oberflächenmodell ist außerdem bezüglich sich intraoperativ ergebenden Veränderungen nicht flexibel genug, da der Abtrag global und prädiktiv berechnet wird und nicht in Echtzeit nach jedem Puls. Das von Stopp [56] vorgeschlagene System beinhaltet ein Planungssystem, sowie eine intraoperative Abtragsberechnung und -visualisierung, welche parallel zur manuell durchzuführenden Laserosteotomie abläuft. Das Abtragsmodell ist voxelbasiert und wird in Echtzeit nach jedem Puls neu berechnet, jedoch beschränkt sich das System auf das Lasern von Kavitäten im Kiefer und es wird kein Roboter verwendet. Daher lässt sich das Verfahren nicht ohne weiteres auf das Lasern von Schnittlinien übertragen, wofür eine höhere Genauigkeit in der Schnittführung nötig ist, als per Hand erreichbar wäre. Die Modellierung des Knochenvolumens mit Voxeln ist wegen der begrenzten Rechenleistung heutiger Computer auf kleine Volumina beschränkt, größere Knochenstrukturen, wie ein Femur, sind damit nicht modellierbar.

Im Rahmen dieser Arbeit soll ein System entwickelt werden, welches es ermöglicht präoperativ den Eingriff einer robotergestützten Laserosteotomie zu planen. Die Planung soll sich an bestehenden Operationsabläufen orientieren und in diese integrierbar sein. Der Prozess des Knochenabtrags soll präoperativ und intraoperativ simulierbar und visualisierbar sein. Im Speziellen soll das System Folgendes ermöglichen:

- Laden dreidimensionaler Anatomiedaten, auch größere Strukturen wie ganze Knochen sollen möglich sein
- Einzeichnen von Schnitttrajektorien beliebiger Komplexität (gerade oder geschwungene Formen) an den Anatomiedaten, welche vom Roboter abgefahren werden können
- Präoperative Simulation und Visualisierung des Schnittprozesses
- Intraoperative Adaption der Planung sowie die Möglichkeit einer Berechnung und Visualisierung des Schnittprozesses parallel zur realen Operation

Dazu muss ein mathematisches Modell entwickelt werden, welches den Knochenabtrag ausreichend genau beschreibt und in Echtzeit berechenbar ist. Das abzutragende Knochenvolumen muss so modelliert werden, dass verschiedene Knocheneigenschaften wie Dichte und Absorptionskoeffizient (z.B. Kompakta und Spongiosa) sowie eventuelle Hohlräume im Knochen berücksichtigt werden können. Die Volumenmodellierung soll räumlich nicht auf den direkten Abtragsbereich begrenzt sein, sondern es sollen auch große Knochenstrukturen wie die eines Femurs modelliert werden können, um flexibel auf intraoperative Änderungen wie das Verschieben einer Schnitttrajektorie reagieren zu können. Außerdem müssen im Modell die laserspezifischen Schnitteigenschaften wie der gaußsche Energieeintrag und das Entstehen von Debris in Betracht gezogen werden. Als Roboter wird dazu der am DLR entwickelte Leichtbauroboter MIRO verwendet, die Planung soll in ein bestehendes Planungssystem für minimalinvasive Eingriffe mit dem MIRO integriert werden. Das System wird am Beispiel der Er:YAG Lasers AT Fidelis von Fotona entwickelt, soll aber flexibel auf andere Lasertypen erweitert werden können. Ziel der Entwicklung dieses Systems ist es, die genannten Vorteile der Laserosteotomie zu nutzen und damit langfristig bessere Behandlungsmethoden zu ermöglichen.

Kapitel 4

Aufbau des Planungssystems

4.1 Gesamtsystem

In diesem Kapitel wird das Gesamtsystem statisch und dynamisch erklärt. Die statische Systembeschreibung definiert die Rolle des Planungssystems im Operationsszenario, sowie die Interaktion mit anderen Komponenten des Operationsszenarios, z.B. dem chirurgischen Laser und dem Roboter. Letztere werden im Abschnitt 4.1.2 genauer beschrieben. Außerdem stellt die statische Systembeschreibung die einzelnen Komponenten des Planungssystems untereinander in Beziehung und identifiziert Schnittstellen. In der dynamischen Systembeschreibung wird der Arbeitsablauf der Planung erklärt.

4.1.1 Statische Systembeschreibung

Das Planungssystem fügt sich, wie in Abbildung 4.1 zu sehen ist, in das gesamte Operationsszenario ein. Präoperativ kann damit in einer simulierten Umgebung die Laserosteotomie geplant und simuliert werden. Dabei können alle wichtigen Komponenten der realen Umgebung simuliert werden, wie z.B.:

- Der Roboter unter Berücksichtigung von Geometrie und Kinematik
- Die anatomischen 3D Daten des Patienten, insbesondere die Knochen, an welchen die Laserosteotomie durchgeführt werden soll, unter Berücksichtigung von Geometrie und Material
- Der Laser unter Berücksichtigung der Laserparameter
- Der Operationsraum unter Berücksichtigung der Raumaufteilung

Die aus der Planung resultierenden Daten, welche die Robotertrajektorie sowie Steuerparameter für den Laser beinhalten, können an das reale System übertragen werden, um den geplanten Eingriff durchzuführen. Bei Durchführung der Laserosteotomie kann der Abtrag parallel und online berechnet sowie visualisiert werden.

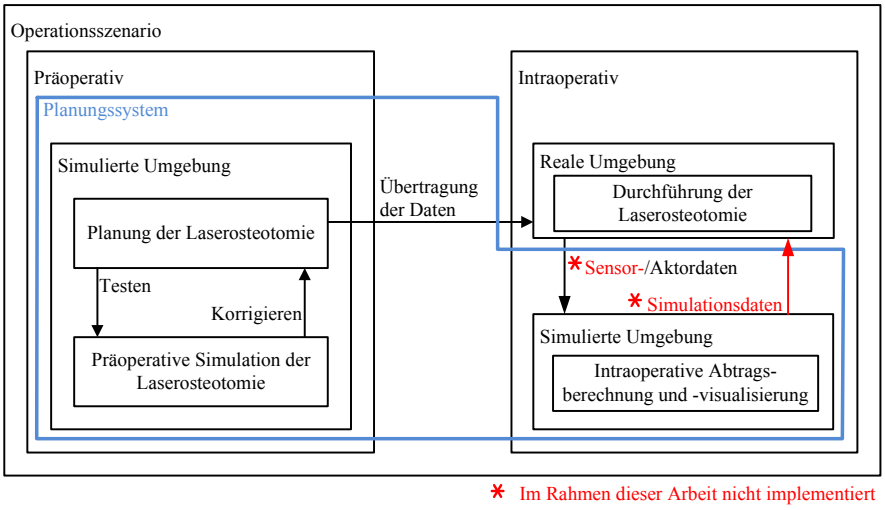


Abbildung 4.1: Einordung des Planungssystems in das Operationsszenario.

Wie in Abbildung 4.2 zu sehen ist, kann das Planungssystem intern in folgende Module unterteilt werden:

- Benutzerschnittstelle
- Externe Software Amira
- Visualisierung
- Modul zur Berechnung der physikalischen Prozesse und zur Simulation

Die grafische Benutzerschnittstelle leitet den Benutzer durch die einzelnen Arbeitsschritte des Planungsprozesses, ermöglicht Benutzereingaben, zeigt Daten an und ist mit allen anderen Modulen verbunden.

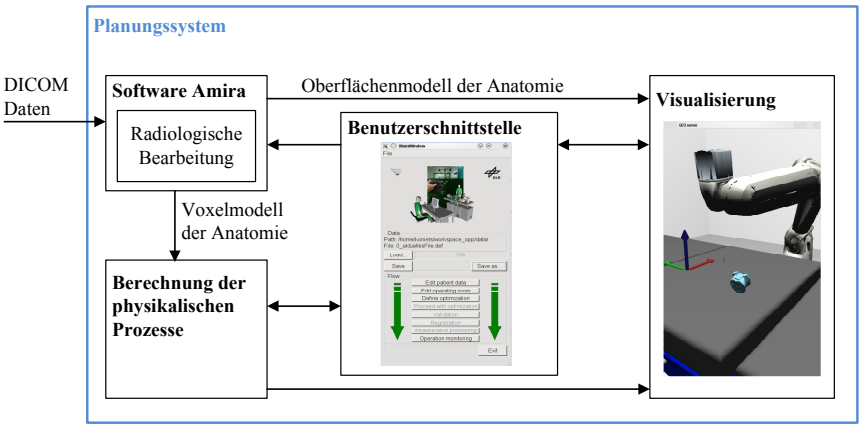


Abbildung 4.2: Interner Aufbau des Planungssystems.

Die Benutzerschnittstelle setzt sich aus den folgende Qt-GUIs (Qt: Offene C++ Bibliothek zur Erstellung von GUIs, GUI: Grafical User Interfaces) zusammen (vgl. Abbildung 4.3):

- Main GUI
- OP Setting GUI
- Laserplanning GUI

Die Main GUI sowie die OP Setting GUI konnten von einem bestehenden System [32] übernommen werden, die Laserplanning GUI wurde im Rahmen dieser Arbeit implementiert. Die OP Setting GUI und die Laserplanning GUI können von der Main GUI erreicht werden. In der OP Setting GUI kann der Operationsraum eingerichtet oder modifiziert werden (vgl. Abschnitt 4.3), in der Laserplanning GUI wird der Schnitt geplant und die Laserosteotomie simuliert (vgl. Abschnitt 4.4 und 4.6). Von der Main GUI wird auch die externe Software Amira gestartet, in welcher die Vorverarbeitung der anatomischen 3D Modelle geschieht.

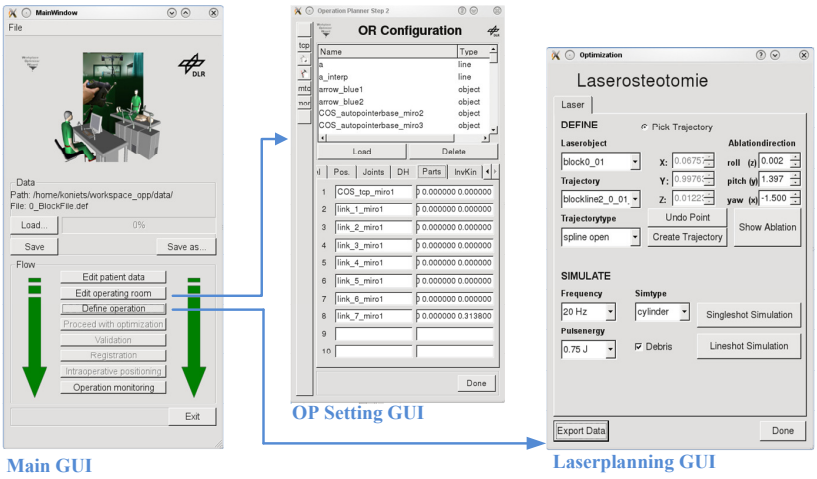


Abbildung 4.3: GUIs zur Realisierung der Benutzerschnittstelle.

Die Visualisierung geschieht mithilfe des auf OpenGL (Open Grafics Library: Programmierschnittstelle zur Entwicklung von Computergrafiken) basierenden Viewers Geoserver. In diesem werden Oberflächenmodelle aller in der Planung verwendeten Komponenten des realen Systems, wie der Roboter, der OP-Saal, sowie die anatomischen Strukturen visualisiert. Es besteht eine bidirektionale Verbindung zur Benutzerschnittstelle. Der Benutzer kann somit nicht nur Oberflächenmodelle in die Visualisierung laden, sondern in dieser auch navigieren und Punkte auf bestimmten Oberflächen markieren und für die spätere Berechnung nutzen. Das Modul zur Berechnung beinhaltet physikalische Modelle der verwendeten Komponenten des realen Systems, wie z.B. das Kinematikmodell des Roboters oder das Materialmodell des Knochens. Es stellt außerdem Berechnungsmethoden bereit, z.B. die Umrechnung kartesischer Punkte in Robotergelenkwinkel über Inverskinematiken oder das physikalische Modell des Lasermaterialabtrags. Über die Benutzerschnittstelle werden die Methoden aufgerufen und Ergebnisse werden an diese zurückgegeben. Manche Methoden beinhalten Visualisierungen, welche im Visualisierungsmodul grafisch dargestellt werden. Das Modul zur Berechnung wurde in C++

programmiert. Näheres zum Aufbau der Software der einzelnen Module kann in Anhang A nachvollzogen werden.

4.1.2 Verwendete Komponenten

Da das Planungssystem, wie in Abbildung 4.1 zu sehen ist, die Daten der Planung an die Roboter- und Lasersteuerung exportiert und diese auch im Berechnungsmodul des Planungssystems modelliert werden, werden sie hier kurz beschrieben.

Operationsroboter MIRO Bei dem in Abbildung 4.4 dargestellten Operationsroboter handelt es sich um den chirurgischen Leichtbauroboter MIRO. Der MIRO stellt nach dem 2006 entwickelten KineMedic die zweite Generation der am DLR entwickelten chirurgischen Leichtbauroboter dar. Er wurde mit dem Ziel entwickelt für vielseitige Zwecke der robotergestützten Chirurgie einsetzbar zu sein, wie die Laserosteotomie oder die minimalinvasive Chirurgie [22]. Bei der Laserosteotomie verfährt der Roboter selbstständig mit einem integrierten Interpolator und stellt somit einen teilautonomen Roboter dar. Bei minimalinvasiven Eingriffen wird er über Force-Feedback Joysticks vom Chirurgen gelenkt, womit er als teleoperativer Roboter fungiert. Um diesen universellen Einsatzmöglichkeiten gerecht zu werden, wird eine redundante Roboterkinematik verwendet, die für eine Vielzahl von operativen Eingriffen optimiert ist. Neben der Vielseitigkeit des Einsatzes stand bei der Entwicklung die Akzeptanz des Anwenders sowie eine sichere Mensch-Maschine-Interaktion im Vordergrund. Daher wurde eine schlanke Leichtbauweise verwendet und es kommt an jedem Gelenk interne Kraft-Momenten-Sensorik zum Einsatz, welche den Roboter Kollisionen erkennen lässt und somit für die nötige Sicherheit bei der direkten Zusammenarbeit von Mensch und Roboter sorgt. Die wichtigsten Parameter des MIRO sind in Tabelle 4.1 zusammengefasst.

Parameter	Wert	Einheit
Gelenkeanzahl	7	$[-]$
Masse	10	$[kg]$
Max. Payload	3	$[kg]$
Kartesische Geschw.	$0,5$	$[m/sec]$
Kontrollzyklus	3	$[kHz]$
Kontrollmodi	Position, Moment, Impedanz	$[-]$
Positioniergenauigkeit	$0,5$	$[mm]$
Wiederholgenauigkeit	$0,2$	$[mm]$

Tabelle 4.1: Parameter des Roboters MIRO, entnommen aus [22], [31].

Die sieben Freiheitsgrade des MIRO werden mit rotatorischen Gelenken realisiert. Mit Gelenk 1 ist eine Drehbewegung des Schultergelenks Roll (Roll-Pitch-Yaw-Winkel sind eine Möglichkeit zur Beschreibung der Werkzeugorientierung eines Roboters) möglich, darauffolgend ermöglicht ein Koppelgelenk die Drehungen Pitch und Yaw um die Achsen 2 und 3 des Schultergelenks. Auch das Ellbogengelenk (Pitch-Roll) mit den Achsen 4 und 5 und das Handgelenk (Pitch-Roll) mit den Achsen 6 und 7 sind als Koppelgelenk ausgeführt. Koppelgelenke haben den Vorteil,

dass aufgrund der Nähe zweier Gelenke elektronische Komponenten reduziert werden können. Die Kinematik ist Anhand der DH-Parameter (DH: Denavit Hartenberg) in Tabelle 4.2 und in Abbildung 4.4 verdeutlicht.

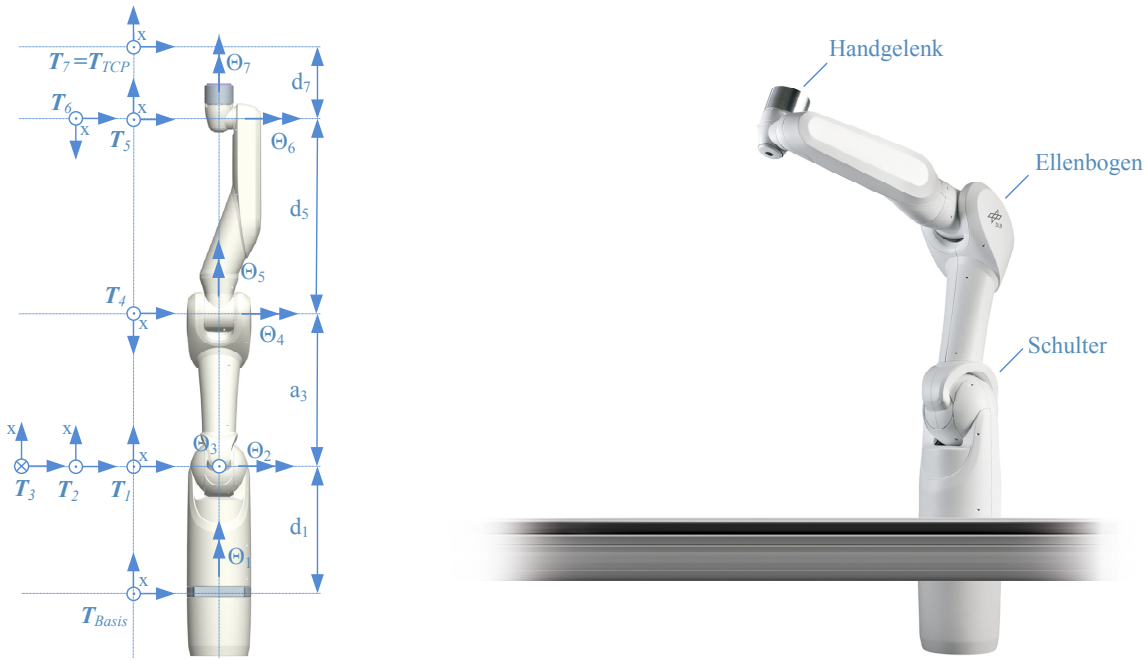


Abbildung 4.4: Der Operationsroboter MIRO. Links: Geometrische Auslegung. Rechts: Anbringung am Operationstisch.

Gelenk i	a_{i-1} $[mm]$	α_{i-1} $[rad]$	d $[mm]$	Θ_i $[rad]$
1	0	0	280	0
2	0	$-\pi/2$	0	$-\pi/2$
3	0	$\pi/2$	0	0
4	310	$-\pi/2$	0	$\pi/2$
5	0	$\pi/2$	385	0
6	0	$-\pi/2$	0	0
7	0	$\pi/2$	200	0

Tabelle 4.2: Die DH-Parameter des MIRO.

Die Ansteuerung des MIRO richtet sich nach der Applikation. Meist wird nach dem Prinzip der Modellgetriebenen Softwareentwicklung (engl.: MDSD - Model-Driven Software Development) vorgegangen, bei dem mithilfe des Realtime Workshop von Matlab/Simulink direkt aus Simulink-Modellen ausführbarer Code gebildet wird. Die Rechnerstruktur einer Ein-Arm-Applikation mit dem MIRO kann z.B. wie in Abbildung 4.5 zu sehen ist, aufgebaut sein. Der aus einem Simulink-Modell abgeleitete Code für die Steuerung und Regelung läuft auf einem QNX-Rechner (QNX: Echtzeitbetriebssystem) mit einer Frequenz von 3 kHz. Dieser QNX-Rechner kommuniziert über den zeitorientierten und

echtzeitfähigen Feldbus SpaceWire [44] mit der Hardware des MIRO. Der für die Applikation benötigte ausführbare Code läuft auf einem weiteren QNX-Rechner im niedrigeren Takt von 1 kHz. Beide QNX-Rechner kommunizieren über das Protokol aRD (aRD: Agile Robot Development) [8]. Auf einem Linux-Rechner kann die Applikation über eine GUI in Simulink gesteuert werden. Die Kommunikation zwischen der GUI und dem Code der Applikation ist ereignisorientiert. Zusätzlich werden dem Bediener die wichtigsten Werte der Regelung auf einer weiteren Oberfläche, dem MIRO Spy, welcher bei den meisten Applikationen verwendet wird, angezeigt.

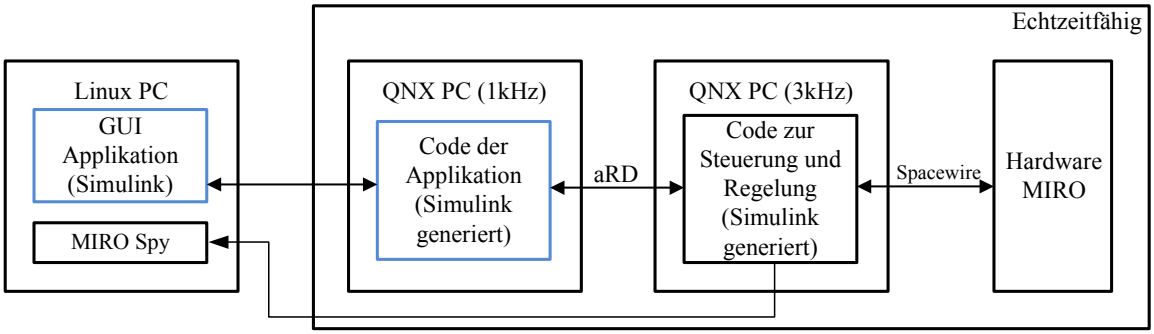


Abbildung 4.5: Mögliche Rechnerstruktur für eine Ein-Arm-Applikation mit dem MIRO.

Chirurgischer Laser AT Fidelis Bei dem verwendeten Laser handelt es sich um den gepulsten Er:YAG Laser AT Fidelis der Firma Fotona, dessen Spezifikationen in Tabelle 4.3 dargestellt sind. Der Er:YAG Laser ist ein Festkörperlaser mit einem mit Erbium dotierter YAG Kristall (Yttrium-Aluminium-Granat) als Lasermedium und einer Blitzlampe als optische Pumpquelle. Die damit erzielbare Wellenlänge von 2.9 μm entspricht der Wellenlänge bei der Wasser und Hydroxyapatit ihr Absorptionsmaximum haben, was den Laser für medizinische Anwendungen, insbesonders für den Hartgewebeabtrag attraktiv macht. Aus dem hohem Absorptionskoeffizienten von Knochen resultiert eine sehr geringe Eindringtiefe, wodurch der Abtrag nahezu ohne thermische Belastung des Gewebes möglich ist [5]. Um den thermischen Eintrag so gering wie möglich zu halten, stellt der verwendete Laser AT Fidelis Modi mit sehr steilen Pulsflanken und kurzen Pulslängen bereit. Die steilen Flanken bewirken, dass mit jedem Puls eine nahezu konstante Energie in den Knochen eingebracht wird und damit über die gesamte Pulslänge der gewünschte Effekt, das explosionsartige Abtragen, erzielt wird.

Parameter	Wert	Einheit
Wellenlänge	$2,94$	$[\mu m]$
Durchschnittliche Leistung	20	$[W]$
Max. Energieflussdichte	48	$[J/cm^2]$
Pulsenergie	$0,02$ - $1,5$	$[J]$
Max. Pulsfrequenz	50	$[Hz]$
Mögl. Pulslängen	$50, 100, 300, 600, 1000$	$[\mu s]$

Tabelle 4.3: Parameter des Lasers AT Fidelis, entnommen aus [17].

Abbildung 4.6: Der chirurgische Laser AT Fidelis von Fotona.

Der AT Fidelis wird hauptsächlich in der Dentalmedizin verwendet. Zur Laserosteotomie findet der Laser vorwiegend im Forschungsbereich Anwendung.

4.1.3 Dynamische Systembeschreibung

Der Arbeitsablauf zur Planung einer Laserosteotomie mit intraoperativer Abtragsberechnung und -visualisierung ist in Abbildung 4.7 zu sehen. Die dabei im Planungssystem erfolgenden Hauptprozesse sind:

- Vorbereitung der anatomischen Daten in Amira
- Präoperative Planung und Simulation
- Intraoperative Abtragsberechnung und -visualisierung

Zuerst müssen anatomische CT- oder MRT-Daten im externen Programm Amira für die Planung vorbereitet werden. Genaueres dazu wird in Abschnitt 4.2 erklärt. Im Hauptprozess Präoperative Planung und Simulation kann zuerst die Operationsumgebung eingerichtet werden. Dabei ist es z.B. möglich, die Position der Roboterbasis am Operationstisch festzulegen, damit der zu bearbeitende Knochen im Arbeitsraum des Roboters liegt. Danach wird das Laserobjekt geladen und es können durch Festlegung von Schnitttrajektorien und -normalen die zu lasernden Schnitte definiert werden. Anschließend kann der Abtrag simuliert werden. Bei zufriedenstellenden Ergebnissen werden die Operationsparameter exportiert, ist dies nicht der Fall, so kann die Planung wiederholt werden. Auf diese Vorgänge wird in den Abschnitten 4.4, 4.5, 4.6 und 4.7 näher eingegangen. Der letzte Prozess, die intraoperative Abtragsberechnung wird parallel zur realen Operation durchgeführt. Dabei werden die realen Prozessdaten in Echtzeit an das Planungssystem gesendet und für eine Berechnung verwendet. Auf den Schritt der intraoperativen Abtragsberechnung wird näher im Abschnitt 4.7 eingegangen.

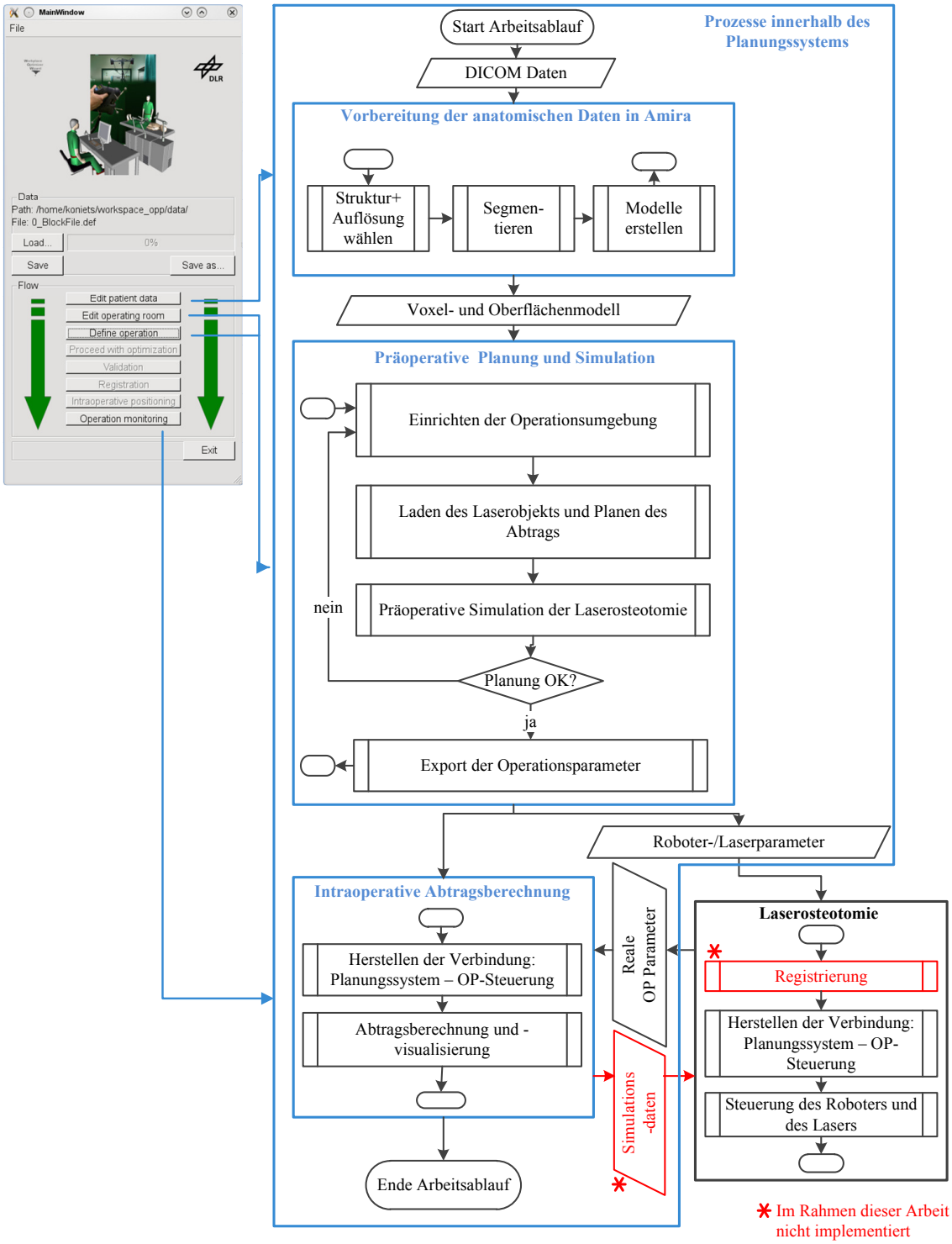
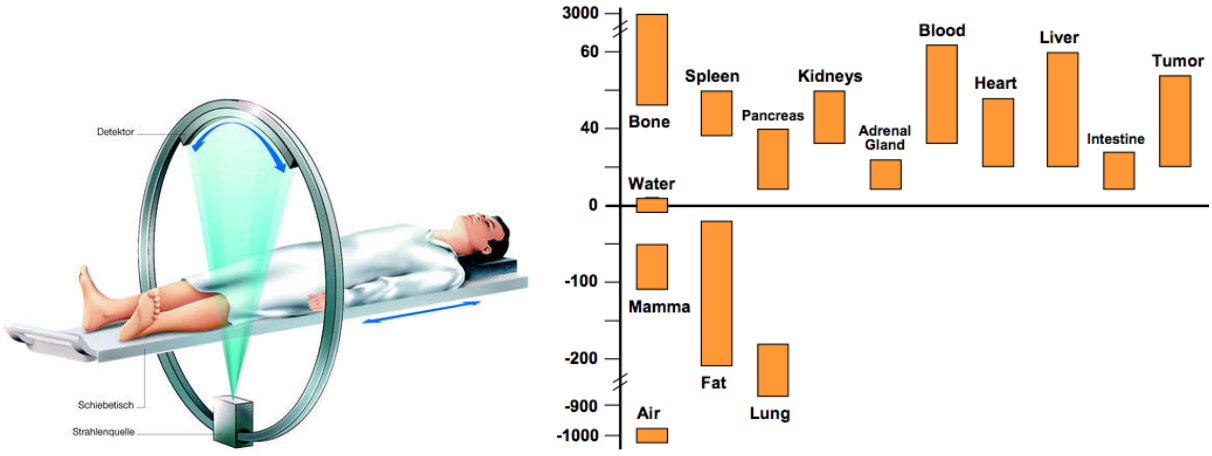


Abbildung 4.7: Arbeitsablauf von der Planung der Laserosteotomie bis zur intraoperativen Abtragsberechnung.

4.2 Vorbereitung der anatomischen Bilddaten

Im ersten Schritt des Arbeitsablaufs (vgl. Abbildung 4.7) werden die 3D Daten bearbeitet. Das Erfassen dieser 3D Daten geschieht im Vorfeld und ist nicht Teil dieser Arbeit. Es soll zum besseren Verständnis hier kurz beschrieben werden.
Zur Erfassung dreidimensionaler anatomischer Daten bestehen verschiedene bildgebende Verfahren. Die Computertomographie (CT) und die Magnetresonanztomographie (MRT) stellen die am meisten verwendeten dar. Erstere basiert auf der Nutzung der 1895 von Wilhelm Conrad Röntgen entdeckten Röntgenstrahlung, welche von einer Strahlenquelle durch Gewebe geleitet und danach auf einem Strahlendetektor abgebildet wird. Je nach Art des Gewebes werden die Strahlen mehr oder weniger absorbiert oder transmittiert, so dass die anatomischen Strukturen sichtbar werden. Wie in Abbildung 4.8(a) zu sehen ist, sind bei der Computertomographie Strahlenquelle und -detektor auf einem rotierenden Bogen gegenübergelegen. Durch dessen Rotation und die kontinuierlicher Aufnahme von Röntgenbilder kann eine Schicht aufgenommen werden. Wird der Patient entlang dieses Bogens bewegt, entsteht aus diesen Schichten eine dreidimensionale Aufnahme der Patientenanatomie. Während bei einer zweidimensionalen Röntgenaufnahme gemäß der Auflösung des Röntgengerätes jedem Flächenelement (Pixel) ein Grauwert zugeordnet wird, geschieht beim CT dasselbe mit jedem Volumenelement (Voxel). Der Grauwert definiert die Absorption bzw. Transmission der Röntgenstrahlung im jeweiligen Voxel, was u.a. einen Rückschluss auf die Dichte der anatomischen Struktur an der Stelle des Voxels zulässt. Der Grauwert wird in Hounsfield angegeben, die dazugehörige Skala (vgl. Abbildung 4.8(b)) reicht von Luft ($\approx -1000\ Hu$), welche die Strahlung nicht absorbiert bis Knochen ($\approx\ 100 - 3000\ Hu$), welcher die Strahlung maximal absorbiert. Der Name der Einheit ist nach dem englischen Elektrotechniker und Mitentwickler des ersten Computertomographen [27] Godfrey Hounsfield benannt.



(a) Aufbau eines CTs (b) Hounsfield Skala [27]

Abbildung 4.8: Funktionsprinzip der Computertomographie.

Die Magnetresonanztomographie basiert auf einer, durch starke Magnetfelder hervorgerufenen, resonanten Anregung bestimmter Atomkerne im Gewebe (meist Wasserstoffkerne), welche bei Relaxation in einem Empfängerstromkreis elektrische Signale induzieren. Der Bildkontrast

zwischen verschiedenen Gewebearten wird dabei u.a. durch unterschiedliche Wasseranteile und damit verbundene Relaxationszeiten definiert [34]. Analog zur CT werden hierbei durch Rotation des Systems um den Patienten dreidimensionale, auf Voxeln basierende Bilddaten gewonnen, deren Grauwerte die verschiedenen Gewebearten definieren. Diese folgen allerdings nicht der Houndsfieldskala. Der Nachteil der CT Technik ist die hohe Strahlenbelastung durch Röntgenstrahlung, der Nachteil der MRT Technik ist, dass sich aufgrund des Magnetfeldes keine metallischen Gegenstände in der Nähe des Systems befinden dürfen. Die Auflösung von kommerziellen CT Systemen in der klinischen Anwendung liegt bei ca. $0,3\ mm$ (Siemens SOMATOM Multicore CT), die von MRT Systemen ist aufgrund technischer Gegebenheiten etwas ungenauer. Näheres zu den verschiedenen bildgebenden Verfahren kann in einschlägiger Literatur nachgelesen werden [40]. Die Bilddaten werden meist als Schichtbilder in dem offenen DICOM-Format (DICOM: Digital Imaging and Communications in Medicine) gespeichert. Zur Analyse und Weiterverarbeitung dieser steht diverse radiologische Software zur Verfügung, u.a. das Open Source Programm Slicer und das kommerzielle Programm Amira, welches in dieser Arbeit verwendet wurde.
Für verschiedene Einsatzbereiche der Software Amira sind unterschiedliche Pakete und Erweiterungen erhältlich, womit die Anwendung von der Segmentierung bis hin zu aufwändigen Renderingverfahren unterschiedlichster radiologischer Daten reicht. Wie in Abbildung 4.9 zu sehen ist, müssen zuerst die CT-Daten im DICOM Format eingelesen werden. Im Rahmen dieser Arbeit lagen Daten eines Rumpfes mit einer Bildauflösung von $0,5\ mm$ und einem Schichtabstand von $0,3\ mm$ vor. Amira interpretiert diese Daten nach dem Einlesen als von einer Boundingbox begrenztes Volumen, welches aus Voxeln der Maße $0,5 \times 0,5 \times 0,3\ mm$ besteht. Um die für die spätere Planung interessante Knochenstruktur auszuwählen kann z.B. in x-, y-, und z-Richtung schichtweise durch das entstandene Voxelvolumen navigiert und die Boundingbox so verschoben werden, dass in dieser nur noch die betroffene Knochenstruktur liegt. Außen liegende Strukturen werden damit verworfen. Danach kann segmentiert werden. Jedes Voxel ist durch seine Größe, seine Position und seinen Hounsfieldwert definiert. Der Hounsfieldwert von Knochen liegt zwischen 100 und $3000\ Hu$ (vgl. Abbildung 4.8), wobei die Spongiosa niedrigere Hounsfieldwerte ausweist als die Kompakta. Um den Knochen aus dem Voxelvolumen zu segmentieren muss ein genauer Wert durch Ausprobieren ermittelt werden. Vorerst wurde auf eine Unterscheidung zwischen Kompakta und Spongiosa verzichtet, woraus sich ein Schwellenwert von $300\ Hu$ ergab. Voxel deren Hounsfieldwerte überhalb von $300\ Hu$ liegen wurden also als Knochen behandelt, Voxel deren Werte darunterlagen wurden entfernt. Daraus ergab sich eine Struktur, wie in Abbildung 4.9 sichtbar ist. Nach der Segmentierung werden die einzelnen Voxel nicht mehr anhand ihrer Hounsfieldwerte definiert, sondern vereinfacht als Knochen (1) oder kein Knochen (0). Amira speichert eine solche segmentierte Struktur als am-Datei (vgl. Anhang B), welches ein Voxelformat ist und folgende Informationen beinhaltet:

- Koordinate zweier Ecken der Boundingbox (vorne-unten-links und hinten-oben-rechts)
- x-, y-, z-Maße der Boundingbox
- Anzahl der Voxel in x-, y- und z-Richtung
- Liste der Voxel, bezeichnet mit 1 (Knochen) und 0 (kein Knochen)

Die Oberfläche der Struktur kann innerhalb von Amira einfach mit Dreiecken angenähert und als stl-Datei exportiert werden, welche die Koordinaten der Dreiecke und zu jedem Dreieck

eine Normale beinhaltet. Damit liegen alle nötigen Informationen über die Oberfläche und das Voxelvolumen der Knochenstruktur vor und die Vorbereitung der 3D Daten in Amira ist abgeschlossen. Die in der am-Datei gespeicherte Voxelinformation kann direkt in das Planungssystem eingelesen werden. Damit die Oberfläche im Geoserver visualisiert werden kann, muss die stl-Datei mithilfe eines C-Programms in eine obj-Datei (vgl. Anhang B) umgewandelt werden.

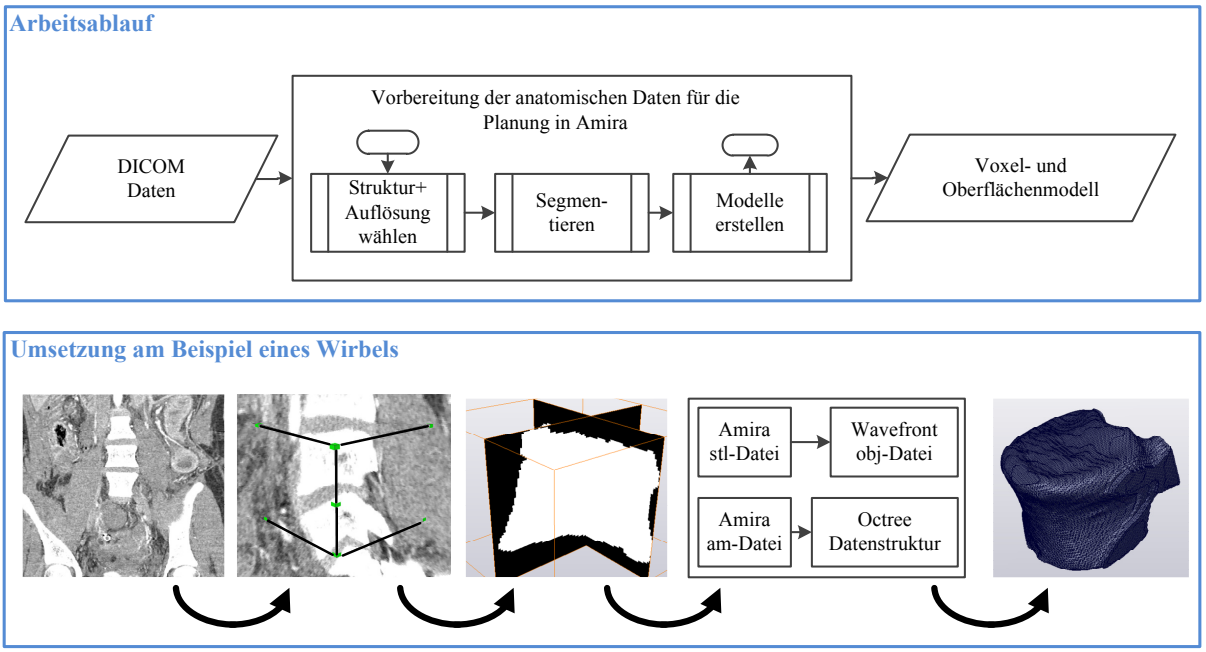


Abbildung 4.9: Arbeitsablauf zur Vorbereitung der 3D Daten.

4.3 Einrichten der Operationsumgebung

Die OP Setting GUI (vgl. Abbildung 4.3) sowie die dafür implementierten Berechnungsmethoden wurden nicht im Rahmen dieser Arbeit entwickelt. Sie konnten von einem bestehenden System zur Planung minimalinvasiver Interventionen mit dem Operationsroboter MIRO [32] übernommen werden. Zum vollständigen Verständnis des Arbeitsablaufes werden sie hier aber erklärt. In diesem Schritt kann der Operationsraum eingerichtet werden, indem Planungskomponenten in das System geladen und modifiziert werden. Die für eine Laserosteotomie wichtigen Komponenten sind (vgl. Abbildung 4.10):

- Roboter
- Trajektorien
- Laserobjekte
- Weitere Objekte wie die OP Liege

Roboter und Laserobjekte haben eigene Koordinatensysteme (${}^{world}\boldsymbol{T}_{RB}$ und ${}^{world}\boldsymbol{T}_{LO}$), welche ihre Lage und Orientierung zum Weltkoordinatensystem $\boldsymbol{T}_{world}$ beschreiben, Trajektorien

werden direkt in Weltkoordinaten angegeben. Alle Komponenten können translatorisch und rotatorisch positioniert werden. Für Roboter sind außerdem folgende weitere Modifikationen möglich:

- Einstellen der DH-Parameter und damit der Position und Lage einzelner Gelenke und Baugruppen zueinander
- Manipulieren der Roboterstellung im Gelenkwinkelraum
- Auswählen einer Inverskinematik
- Manipulieren des Roboter TCPs im kartesischen Raum (über die gewählte Inverskinematik)

Außerdem kann z.B. ein bestimmter Arbeitsbereich für eine Operation gewählt werden und die optimale Positionierung eines oder mehrerer Roboter unter Berücksichtigung von Erreichbarkeit des Arbeitsbereichs und Kollision zwischen den Robotern über generische Algorithmen errechnet werden.

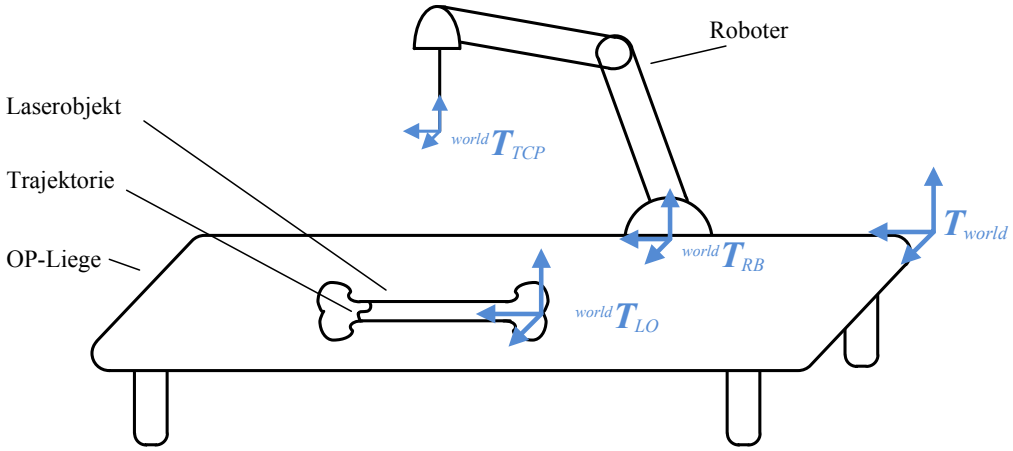


Abbildung 4.10: Räumlicher Aufbau der virtuellen OP Umgebung.

4.4 Volumenmodellierung und Schnittplanung

Im nächsten Schritt des Arbeitsablaufs (vgl. Abbildung 4.7) kann das Laserobjekt geladen und der Schnitt über die Schnitttrajektorie und -normale definiert werden. Diese Einzelschritte kann der Bediener, wie in Abbildung 4.11 zu sehen ist, über die Laserplanning GUI und im Visualisierungsmodul vornehmen. Sie werden im Folgenden erklärt. Dazu muss zuerst eine geeignete Volumenmodellierung für das Laserobjekt gefunden werden.

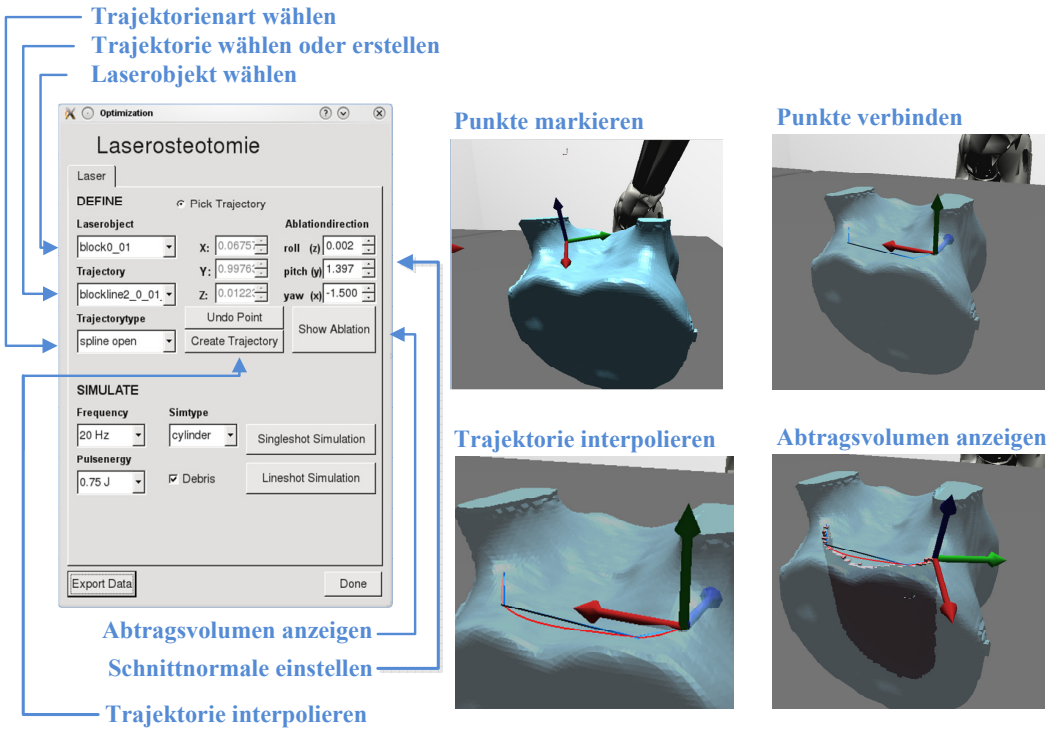


Abbildung 4.11: Einzelschritte Schnittplanung über die Bedienerschnittstelle. Links: Die Laserplanning GUI. Rechts: Visualisierung der Einzelschritte im Geoserver.

4.4.1 Volumenmodellierung

Nach der Vorbereitung der 3D Daten mit Amira stehen zur Beschreibung des Laserobjekts eine obj-Datei mit der Oberflächeninformation und eine am-Datei mit der Information über das Voxelvolumen zur Verfügung. Mithilfe dieser muss das Laserobjekt unter folgenden Anforderungen modelliert werden:

- Visualisierung des gesamten Laserobjekts im Geoserver
- Visualisierung einzelner Voxel im Geoserver (zur Visualisierung des Abtrags)
- Speichern des gesamten Voxelvolumens und Berechnung des Abtrags

Je nach Anforderungen kann Volumen auf verschiedene Arten modelliert werden. Bei industriellen CAD/CAM (Computer Aided Design/Computer Aided Manufacturing) Anwendungen beispielsweise können Volumen über ihre Kanten (Wireframe Modelling), ihre Oberflächen (B-Rep: Boundary Representation Modelling) oder mithilfe parametrisierbarer geometrischer Primitive (CSG: Constructive Solid Geometry Modelling) beschrieben werden [2]. Die drei Möglichkeiten sind in Abbildung 4.12 dargestellt. CSG Modelle benötigen vergleichbar geringe Rechenzeiten und sind echte Volumenmodelle, d.h. jegliche Raumpunkte können anhand ihrer Koordinaten und der analytischen Beschreibung des Volumenkörpers einfach als innerhalb oder außerhalb des Körpers bestimmt werden. Allerdings sind sie in der Komplexität ihrer Geometrie begrenzt und bei jeglicher Änderung des Körpers muss das globale Modell neu berechnet werden. B-Rep Modelle, deren Oberfläche meist über kleine Teilflächen und deren Normale (z.B. Dreiecke bei .stl-Formaten) beschrieben werden (eine weitere Möglichkeit der Oberflächenbeschreibung sind NURBS - Nonlinear Uniform Rational B-Splines), sind bezüglich der Komplexität ihrer Form flexibler. Allerdings entstehen dabei große Datenmengen und hohe Rechenzeiten und die Beschreibung ist keine echte Volumenbeschreibung, d.h. ob ein Punkt innerhalb oder außerhalb des Körpers liegt ist schwer berechenbar.

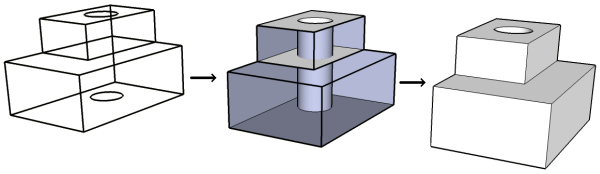


Abbildung 4.12: Verschiedene Arten der Volumenmodellierung. Links: Wireframe Modelling. Mitte: Constructive Solid Geometry Modelling. Rechts: Boundary Representation Modelling.

Anatomische Strukturen, wie die eines Knochens sind im Allgemeinen zu kompliziert um mit CSG Modellen dargestellt werden zu können. Für Visualisierung des gesamten Laserobjekts sowie einzelner Voxel im Geoserver sollen daher B-Rep Modelle verwendet werden. Für die Berechnung des Laserabtrags muss allerdings jeder Punkt des Volumens beschrieben werden um z.B. verschiedene Härtegrade innerhalb des Knochens oder Hohlräume definieren zu können. Dies ist weder mit CSG Modellen noch mit B-Rep Modellen, sondern nur mit Voxelmodellen möglich. Eine solche voxelbasierte Volumendarstellung kann geschehen, indem das ganze Volumen in Voxeln einheitlicher, von der Auflösung bestimmter Größe aufgeteilt wird (NAZ: Normzellen-Aufzählungsschema, engl.: SOE: Spatial occupancy enumeration). Dabei entstehen Datenmengen, die B-Rep Modelle mit ähnlicher Auflösung bei weitem übertreffen. Eine Möglichkeit diese Modelle zu verschlanken sind Oktalbäume bzw. Octrees. Octrees bestehen aus Voxeln unterschiedlicher Größe und fassen Ansammlungen von gleichen Voxeln zu einem großen Voxel zusammen (vgl. Abbildung 4.13).

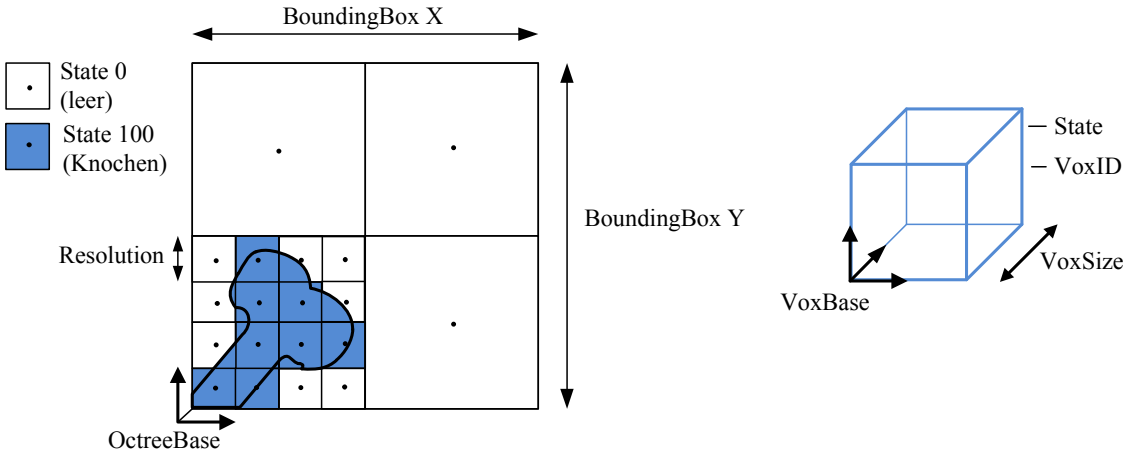


Abbildung 4.13: Links: 2D Ansicht eines Octrees, der einen Knochen enthält mit Attributen. Rechts: Voxel mit Attributen.

Im Berechnungsmodell wurde die Klasse Laserobjekt erstellt, welche zur Erfüllung der drei genannten Anforderungen folgende Attribute besitzt:

- Geoserver-Oberflächenobjekt zur Visualisierung des gesamten Laserobjekts (B-Rep)
- Geoserver-Voxelobjekt zur Visualisierung einzelner Voxel (B-Rep)
- Volumenobjekt zum Speichern des gesamten Voxelvolumens und zur Berechnung des Abtrags (Octreestruktur)

Das Geoserver-Oberflächenobjekt wird bei Initialisierung des Planungssystems aus der obj-Datei erstellt und im Geoserver visualisiert. Es dient rein zur Darstellung des gesamten Objekts im Geoserver und zum Kennzeichnen von Stützpunkten der zu planenden Trajektorie. Das Geoserver-Voxelobjekt wird wie das Geoserver-Oberflächenobjekt bei der Initialisierung erstellt, ist aber leer, solange keine Voxel des Laserobjekts dargestellt werden sollen. Es dient nur zum Visualisieren einzelner Voxel im Objekt. Ein so visualisiertes Voxel enthält dabei keine echte Volumeninformation, wie die Dichte, sondern stellt nur seine acht Oberflächen im Geoserver dar. Die Octreestruktur wird erst erstellt, wenn dieses Laserobjekt über die Laserplanning GUI zur Bearbeitung ausgewählt wurde. Es dient zur Modellierung der Volumeneigenschaften und zur späteren Berechnung des Laserabtrags und wird aus der am-Datei erstellt. Die Octreestruktur entspricht einer eigenen Klasse, welche aus einer am DLR entwickelten [6] C++ Bibliothek übernommen werden konnte. Diese Klasse Octree und ihre für diese Arbeit wichtigsten Methoden und Eigenschaften können wie folgt beschrieben werden (vgl. Abbildung 4.13).

- Der Octree wird durch eine Basis, eine ihn umgrenzende Bounding Box sowie eine Auflösung definiert. Die Auflösung entspricht der Kantenlänge des kleinsten Voxels.
  → Attribut **OctreeBase**
  → Attribut **BoundingBox**
  → Attribut **Resolution**

- Jedes Octreevoxel ist über eine eindeutige ID, eine kartesische Basis und eine Kantenlänge definiert.
  → Attribut **VoxID**
  → Attribut **VoxBase**
  → Attribut **VoxSize**

- Der Octree wird gefüllt indem einzelnen kartesischen Punkten eine Eigenschaft zugewiesen wird. Abhängig von der Auflösung wird das Voxel, in welchem der jeweilige Punkt liegt, mit dieser Eigenschaft belegt. Innerhalb des Voxels ist dann jedem kartesischen Punkt die entsprechende Eigenschaft zugewiesen.
  → Attribut **State**
  → Methode **setState()**
  → Methode **getState()**

- Die Voxelanzahl im Octree verhält sich dabei dynamisch. Wenn jeder Punkt innerhalb der Bounding Box dieselbe Eigenschaft aufweist besteht der Octree unter Umständen nur aus einem Voxel. Bei vielen verschiedenen Eigenschaften steigt die Anzahl der Voxel.

- Die Klasse Octree stellt eine Raytracingmethode bereit, welche die von einem Strahl getroffenen Voxel finden kann. Der Strahl ist durch eine kartesische Basis und eine Richtung definiert.
  → Attribut **Beam**
  → Methode **getBeamIntersectingPoints()**

- Die Klasse Octree stellt eine Methode bereit, welche Octreevoxel finden kann, die sich mit einem anderen Kubus überschneiden. Der Kubus ist durch eine kartesische Basis und eine Kantenlänge definiert.
  → Attribut **Cube**
  → Methode **getIntersectingObjects()**

Sobald ein Laserobjekt ausgewählt wurde, wird dieses mit setState() gesetzt. Die dazu nötigen Daten werden der am-Datei entnommen. Dort sind die Maße der Bounding Box, die Auflösung sowie zu jedem Voxel eine binäre Eigenschaft (0: kein Knochen, 1: Knochen) gespeichert. Ein Voxel, welches kein Knochenmaterial beinhaltet, wird mit setstate(0) gesetzt, ein Voxel, welches Knochenmaterial beinhaltet wird mit setstate(100) gesetzt. Die 100 steht für einen Gesundheitszustand $H \in [0; 100]$, und bedeutet, dass jedes Voxel vorerst einen Gesundheitszustand von $100\ \%$ besitzt. Bei der späteren Abtragsberechnung wird dieser Gesundheitszustand entsprechend verringert, sobald Laserenergie in das Voxel eingebracht wird. Bei einem Energiezustand von $0\ \%$ gilt das Voxel als eliminiert. Näheres dazu wird in Abschnitt 4.6 erklärt.

4.4.2 Schnittplanung

Zur Definition eines Schnitts kann der Bediener im Geoserver zu dem gewählten Laserobjekt navigieren. Über die mittlere Maustaste lassen sich auf dessen Oberfläche Punkte markieren. Wenn zuvor in der Laserplanning GUI eine bestehende Trajektorie ausgewählt oder eine neue erstellt wurde, werden die markierten Punkte dieser hinzugefügt. Punkt für Punkt kann so die später zu lasernde Trajektorie aufgebaut werden. Bei jedem markierten Punkt wird in der Laserplanning GUI automatisch eine Schnittnormale vorgeschlagen, die der Normalen des Punkts auf der Laserobjektoberfläche entspricht (vgl. Abbildung 4.11). Diese kann auch manuell über den Roll-, Pitch- und Yawwinkel eingestellt werden. Somit sind vorerst nur einzelne TCPs mit Position und Orientierung und die sie verbindenden Geraden festgelegt. Damit diese vom Roboter abgefahren werden können, muss berücksichtigt werden, wie dieser Trajektorien fahren kann. Dafür sind unter anderem folgende Modi bekannt:

- Point-to-Point Bewegung: der Roboter fährt mit dem TCP die gegebenen Stützpunkte an, die Streckenstücke dazwischen sind nicht vorhersehbar und werden gemäß der für den Roboter optimalen Gelenkwinkelkonfigrationen berechnet. Die Geschwindigkeiten an den Stützpunkten sind Null.
- Lineare Bewegung: Der Roboter fährt mit dem TCP die gegebenen Stützpunkte an, wobei der TCP auf einer linearen Bahn verfährt. Die Geschwindigkeiten an den Stützpunkten sind Null.
- Spline Bewegung: Der Roboter fährt mit dem TCP die gegebenen Stützpunkte an, die Streckenstücke dazwischen werden so interpoliert, dass die Geschwindigkeiten über die ganze Trajektorie stetig und differenzierbar ist. Die Geschwindigkeiten an den Stützpunkten sind nicht Null.

Bei der Laserosteotomie verwendete Trajektorien sollen vorhersehbar sein, d.h. die geplanten sollen den später vom Roboter abgefahrenen Trajektorien entsprechen. Mit der Point-to-Point Bewegungen ist dies nicht möglich. Bezüglich ihrer Form sollen lineare Bewegungen (z.B. für zackige Trajektorien) und splineförmige Bewegungen (z.B. für geschwungene Trajektorien) möglich sein. Bei beiden Modi muss zur Ansteuerung des Roboters ein kartesischer Interpolator eingesetzt werden. Dabei wird bei linearen Bewegungen nur von Punkt zu Punkt, bei splineförmigen Bewegungen von einem Startpunkt über Stützpunkte zu einem Endpunkt interpoliert. Um sicherzustellen, dass die geplante Trajektorie der später vom Roboter abzufahrenden entspricht, ist es sinnvoll zur Generierung der im Geoserver dargestellten Trajektorie den Interpolator zu verwenden, welcher später auch zur Ansteuerung des Roboters verwendet wird. In dieser Arbeit soll hierfür die kubische Interpolation verwendet werden, da diese Methode wenig Rechenzeit benötigt [61] und eine stetige Position, Geschwindigkeit und Beschleunigung gewährleistet. Da bei der Laserosteotomie keine mechanische Interaktion zwischen TCP und dem zu schneidenden Knochen stattfindet, ist die Stetigkeit des Ruckes, wofür ein Interpolator höherer Ordnung nötig wäre, nicht zwingend. Die zuvor festgelegten Punkte können so in der Laserplanning GUI über Betätigung eines Knopfes (vgl. Abbildung 4.11) interpoliert und direkt dargestellt werden. Damit ist sichergestellt, dass die vom Bediener geplanten und im Geoserver visualisierten Trajektorien auch vom Roboter gefahren werden können. Die mathematische Beschreibung sowie die verwendete Implementierung des Interpolators werden hier kurz erklärt.

Interpolator Die interpolierte Trajektorie wird durch ihre Stützpunkte definiert, also diejenigen Punkte, welche der Bediener im Geoserver markieren kann. Bei der mathematischen Beschreibung des kubischen Interpolators nach Craig [13] werden diese in Start- und Endpunkt, sowie die dazwischenliegenden Stützpunkte aufgeteilt. Die Geschwindigkeit am Start- und Endpunkt muss bekannt sein, im Normalfall ist sie Null. Die Interpolation berechnet dann für jeden der sechs Raumfreiheitsgrade Θ_i mit $i \in \{0; 1; ..; 5\}$ eine Bahn, die vom Start- zum Endpunkt durch die Stützpunkte führt und deren Geschwindigkeit stetig ist. Da der Bediener die Trajektorie im kartesischen Raum einzeichnen und planen möchte, wird ein kartesischer Interpolator verwendet. Θ_i entsprechen also drei Rotationen sowie drei Translationen im Raum. In Abbildung 4.14 ist eine Interpolierte mit einem Startpunkt P_0, den Stützpunkten P_1 und P_2 und einem Endpunkt P_3 zu sehen. Die Freiheitsgrade Θ_i legen in jedem Punkt P_j mit $j \in \{0; 1; 2; 3\}$ ein Koordinatensystem fest, welches dem Koordinatensystem des Roboter TCPs $\boldsymbol{T}_{TCP,j}$ im jeweiligen Punkt entspricht. In der hier gewünschten Anwendung ist die Orientierung einer Achse des TCPs durch die Schnittnormale $\boldsymbol{n}$ festgelegt, somit entfallen zwei rotatorische Freiheitsgrade Θ_i. Da der Laser ein rotationssymmetrisches Werkzeug ist, ist der dritte rotatorische Freiheitsgrad frei wählbar. In der hier getätigten Implementierung wird er aber so festgelegt, dass die Orientierung des Koordinatensystems $\boldsymbol{T}_{TCP,j}$ über die gesamte Interpolierte konstant bleibt. Daher muss nur über die drei translatorischen Freiheitsgrade Θ_i mit $i \in \{0; 1; 2\}$ interpoliert werden.

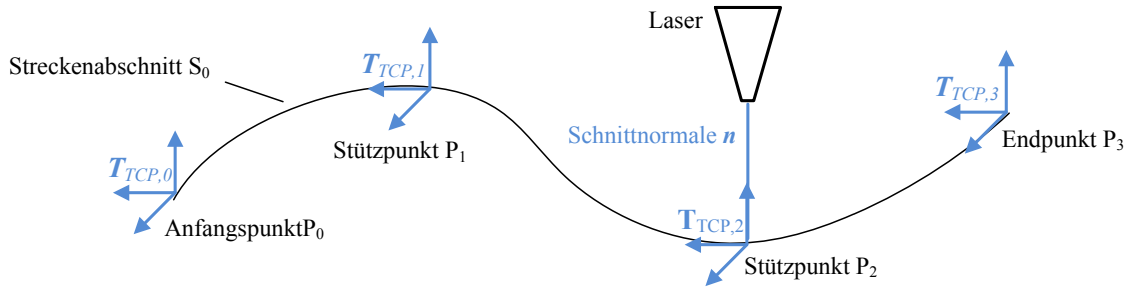


Abbildung 4.14: Eine durch den Start- und Endpunkt sowie zwei Stützpunkte definierte Interpolation.

Der Roboter arbeitet im Gelenkwinkelraum, weshalb zur seiner Ansteuerung noch eine Inverskinematik notwendig ist. Diese wird hier nicht näher erklärt, da sie nicht Teil dieser Arbeit ist, sondern unverändert übernommen wurde. Näheres zur verwendeten Inverskinematik kann in [32] nachvollzogen werden. Nach [13] wird jeder zu interpolierende Freiheitsgrad Θ_i im Streckenabschnitt S_j zwischen zwei Punkten P_j und P_{j+1} durch ein Polynom dritten Grades $\Theta_{i,j}(t)$ (in den folgenden Gleichungen werden die Indizes i und j weggelassen) mit dem Laufparameter $t = [0; t_f]$ definiert. Bei $t = 0$ sowie $t = t_f$ sind das Polynom und dessen erste Ableitung bekannt, so dass gilt:

$$\Theta(0) = \Theta_0 \tag{4.1}$$

$$\dot{\Theta}(0) = \dot{\Theta}_0 \tag{4.2}$$

$$\Theta(t_f) = \Theta_{t_f} \tag{4.3}$$

$$\dot{\Theta}(t_f) = \dot{\Theta}_{t_f} \tag{4.4}$$

Das Polynom kann dann durch die Gleichungen 4.5 bis 4.8 bestimmt werden:

$$\Theta_0 = a_0 \tag{4.5}$$

$$\Theta_{t_f} = a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 \tag{4.6}$$

$$\dot{\Theta}_0 = a_1 \tag{4.7}$$

$$\dot{\Theta}_{t_f} = a_1 + 2a_2 t_f + 3a_3 t_f^2 \tag{4.8}$$

Bei Auflösung der Gleichungen 4.5 - 4.8 nach a_k mit $k \in \{0; 1; 2; 3\}$ resultiert folgendes Gleichungssystem:

$$a_0 = \Theta_0 \tag{4.9}$$

$$a_1 = \dot{\Theta}_0 \tag{4.10}$$

$$a_2 = \frac{3}{t_f^2}\left(\Theta_{t_f} - \Theta_0\right) - \frac{2}{t_f}\dot{\Theta}_0 - \frac{1}{t_f}\dot{\Theta}_{t_f} \tag{4.11}$$

$$a_3 = -\frac{2}{t_f^3}\left(\Theta_{t_f} - \Theta_0\right) + \frac{1}{t_f^2}\left(\dot{\Theta}_{t_f} + \dot{\Theta}_0\right) \tag{4.12}$$

Mit den Gleichungen 4.9 bis 4.12 kann für jeden Streckenabschnitt S_j das entsprechende Polynom bestimmt werden. Während die Geschwindigkeit am Start- und Endpunkt vom Bediener festgelegt wird, gibt es verschiedene Möglichkeiten diese an den Stützpunkten zu bestimmen. Entweder sie werden auch vom Bediener bestimmt, sie werden vom System gemäß einer festgelegten heuristischen Regel bestimmt oder das System berechnet sie so, dass an den Stützpunkten auch die Beschleunigung stetig ist. Bei dem in dieser Arbeit verwendeten Interpolator wird letztere Option gewählt, da so eine gleichmäßigere Bahn entsteht. Soll dies z.B. in einer Weise geschehen, dass ein Freiheitsgrad $\Theta(t)$ von einem Startpunkt Θ_0 über einen Stützpunkt Θ_v bis zu einen Endpunkt Θ_g interpoliert wird, so werden die zwei entstehenden Streckenabschnitte S_0 und S_1 durch die Polynome $\Theta_0(t)$ und $\Theta_1(t)$ definiert:

$$\Theta_0(t) = a_{00} + a_{01} t + a_{02} t^2 + a_{03} t^3 \tag{4.13}$$

$$\Theta_1(t) = a_{10} + a_{11} t + a_{12} t^2 + a_{13} t^3 \tag{4.14}$$

Beide Polynome j werden jeweils von $t = [0; t_{jf}]$ mit $j \in \{0; 1\}$ beschrieben. Vier Randbedingungen ergeben sich dabei aus dem Freiheitsgrad selbst, drei aus dessen Geschwindigkeit und eine aus dessen Beschleunigung.

$$\Theta_0 = a_{00} \tag{4.15}$$

$$\Theta_v = a_{00} + a_{01}t_{f0} + a_{02}t_{f0}^2 + a_{03}t_{f0}^3 \tag{4.16}$$

$$\Theta_v = a_{10} \tag{4.17}$$

$$\Theta_g = a_{10} + a_{11}t_{f1} + a_{12}t_{f1}^2 + a_{13}t_{f1}^3 \tag{4.18}$$

$$0 = a_{01} \tag{4.19}$$

$$0 = a_{11} + 2a_{12}t_{f1} + 3a_{13}t_{f1}^2 \tag{4.20}$$

$$a_{01} + 2a_{02}t_{f0} + 3a_{03}t_{f0}^2 = a_{11} \tag{4.21}$$

$$2a_{02} + 6a_{03}t_{f0} = 2a_{12} \tag{4.22}$$

Die Gleichungen 4.15 bis 4.22 können analog zu den Gleichungen 4.9 bis 4.12 nach a_{jk} mit $k \in \{0; 1; 2; 3\}$ aufgelöst werden, wodurch eine Matrix entsteht. Diese Methode kann auf eine beliebige Anzahl von Stützpunkten ausgebaut und damit eine Trajektorie berechnet werden, deren Geschwindigkeit und Beschleunigung im jeweiligen Freiheitsgrad stetig ist.

Für diese Arbeit konnte auf eine bestehende C++ Implementierung eines Interpolators zurückgegriffen werden, welcher nach den Gleichungen 4.13 - 4.22 von einem Start- bis zu einem Endpunkt eine Trajektorie gemäß Abbildung 4.14 mit maximal 50 Stützpunkten berechnen kann. Dazu werden folgende Eingangsparameter benötigt:

- Taktzeit t_{Takt}
- Θ_i mit $i \in \{0; 1; 2\}$ im Startpunk, im Endpunkt, sowie in den Stützpunkten
- Geschwindigkeit der Freiheitsgrade Θ_i im Startpunkt $\dot{\Theta}_{i,Start}$, sowie im Endpunkt $\dot{\Theta}_{i,End}$
- Maximale Geschwindigkeit $\dot{\Theta}_{i,max}$ der Freiheitsgrade Θ_i auf der Trajektorie

Bei Verwendung des Interpolators gemäß den Gleichungen 4.13 - 4.22 und mit gesetzten $\dot{\Theta}_{i,Start} = 0$ und $\dot{\Theta}_{i,End} = 0$ sind nur offene, splineförmige Trajektorienformen, wie in Abbildung 4.14 zu sehen ist, berechenbar. Dem Bediener sollen aber insgesamt folgende vier Trajektorienarten ermöglicht werden:

- Trajektorienart 1: Lineare Verbindung zwischen den Punkten, offene Trajektorie
- Trajektorienart 2: Lineare Verbindung zwischen den Punkten, geschlossene Trajektorie

- Trajektorienart 3: Splineförmige Verbindung zwischen den Punkten, offene Trajektorie
- Trajektorienart 4: Splineförmige Verbindung zwischen den Punkten, geschlossene Trajektorie

Die bestehende Implementierung musste daher um einen Funktionsblock erweitert werden, um die zusätzlichen Eingabeparameter offen/geschlossen bzw. gerade/splineförmig zu berücksichtigen und alle vier Trajektorienarten ermöglichen zu können (vgl. Abbildung 4.15).

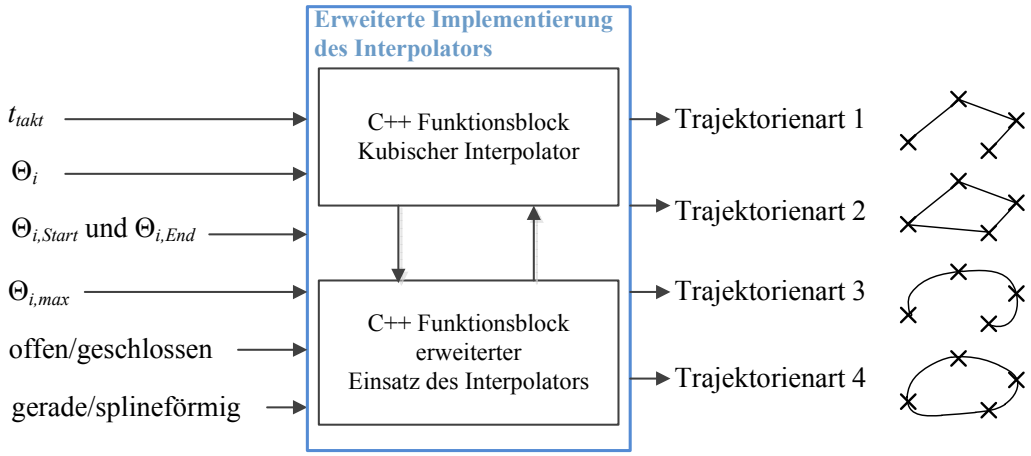


Abbildung 4.15: Implemtierung des Interpolators.

Zur Erzeugung der Trajektorienarten 1 und 2 wird die Trajektorie nicht wie in Abbildung 4.14 zu sehen ist vom Start- bis zum Endpunkt über die Stützpunkte interpoliert. Stattdessen stellt jeder Streckenabschnitt S_j eine Interpolation von $\Theta_{i,j}$ bis $\Theta_{i,j+1}$ ohne Stützpunkte dar. Die Geschwindigkeit an allen Punkten P_j beträgt dabei Null. Trajektorienart 3 erfolgt wie erklärt gemäß Gleichungen 4.13 - 4.22 mit $\dot{\Theta}_{i,Start} = 0$ und $\dot{\Theta}_{i,End} = 0$. Zur Erzeugung von Trajektorienart 4 wird der Startpunkt gleich dem Endpunkt gesetzt, so dass gilt:

$$\Theta_{i,Start} = \Theta_{i,End} \tag{4.23}$$

Wenn an diesen, wie bei Trajektorienart 3 für alle $i \in \{0;1;2\}$ $\dot{\Theta}_{i,Start} = 0$ bzw. $\dot{\Theta}_{i,End} = 0$ gilt, so entsteht in den meisten Fällen am Start- bzw. Endpunkt eine Ecke, wie in Abbildung 4.16 auf der linken Seite zu sehen ist. Auf einer geschlossenen gleichmäßigen Trajektorie darf die absolute Geschwindigkeit des TCPs $|\boldsymbol{v}|$ niemals Null werden, wobei $|\boldsymbol{v}_j|$ bei gleichbleibender Orientierung des TCPs am Punkt P_j folgendermaßen definiert ist:

$$|\boldsymbol{v}_j| = \sqrt{\dot{\Theta}_{0,j}^2 + \dot{\Theta}_{1,j}^2 + \dot{\Theta}_{2,j}^2} \tag{4.24}$$

Eine solche geschlossene Trajektorie ist mit dem implementierten Interpolator z.B. so zu erreichen, indem die Eingabeparameter $\dot{\Theta}_{i,Start}$ sowie $\dot{\Theta}_{i,End}$ gleichgesetzt werden und in mindestens einem der drei Freiheitsgrade nicht Null betragen, damit $|\boldsymbol{v}| \neq 0$ gilt. In diesem Fall müssten $\dot{\Theta}_{i,Start}$ und $\dot{\Theta}_{i,End}$ durch den Bediener eingegeben werden, was zusätzlichen Aufwand

bedeutet. Darüber hinaus entstehen bei unterschiedlichen Eingaben unterschiedliche Trajektorien. Daher wird hier anders vorgegangen. Beginnend mit $\dot{\Theta}_{i,Start} = 0$ wird vom Startpunkt P_{Start} mehrmals über alle Punkte P_j bis zum Endpunkt P_{End} interpoliert, an welchem wieder $\dot{\Theta}_{i,End} = 0$ gilt. Der Startpunkt bzw. Endpunkt wird so zwischenzeitlich auch als Stützpunkt verwendet. In der Mitte der Abbildung 4.16 ist eine solche Interpolation vom Startpunkt $P_{Start} = P_0$ über die Stützpunkte P_{1-5} bis zum Endpunkt $P_{End} = P_6$ zu sehen. Der Anfangsteil von P_0 bis P_1 sowie der Endteil der Trajektorie von P_4 bis P_6 ist hellblau, das Zwischenstück von P_1 bis P_4 ist schwarz gefärbt. Auf dem schwarzen Teil der Trajektorie ist die Geschwindigkeit $|\boldsymbol{v}|$ niemals Null. Dies ist eine vereinfachte Darstellung bei der die Punkte von P_{Start} bis P_{End} nur zweimal umfahren werden.

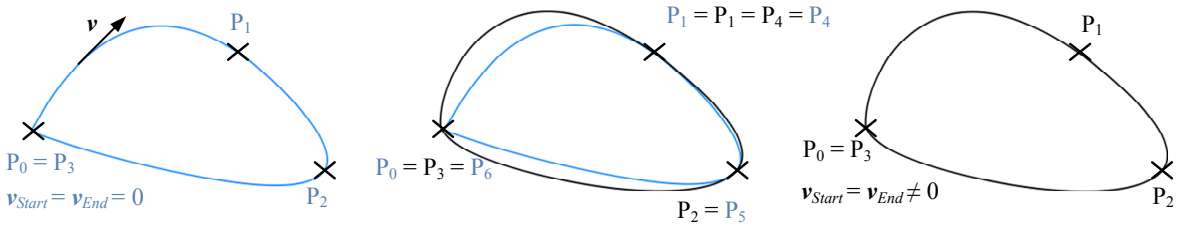


Abbildung 4.16: Erzeugung einer geschlossenen Linie mit dem Interpolator.

Unter Umständen sind mehr als eine Umfahrungen nötig um eine gleichmäßige Bahn zu erhalten. Das Kriterium in der erweiterten Implementierung des Interpolators, ab wann eine Bahn als gleichmäßig angenommen wird ist folgendes: Die n-te Umfahrung bei der die Geschwindigkeit $\boldsymbol{v}_{Start}$ im Punkt P_{Start} der Geschwindigkeit in P_{Start} während der $(n-1)$-ten Umfahrung entspricht wird als gleichmäßige Bahn angenommen. Dem Bediener wird nur diese Umfahrung im Geoserver angezeigt, welche der schwarzen Trajektorie in Abbildung 4.16 auf der rechten Seite entspricht. Der Roboter startet aber immer mit der Geschwindigkeit $|\boldsymbol{v}| = 0$, so dass er, um eine solche Bahn abfahren zu können, an einem bestimmten Punkt in die Bahn einkoppeln muss. Dies wird am Punkt $P_{Start} = P_{End}$ getan. Daher muss die Geschwindigkeit $v_{Start} = v_{End}$ mit der Trajektorie gespeichet werden. Näheres dazu wird in Abschnitt 4.7 erklärt.

Anzeigen des Abtragsvolumens Nachdem eine Trajektorie und eine Schnittnormale bestimmt wurden, kann ein ungefähres Abtragsvolumen angezeigt werden. Dies kann durch Drücken des entsprechenden Knopfes in der Laserplanning GUI geschehen (vgl. Abschnitt 4.11). Durch die Anzeige wird sichtbar durch welche anatomischen Strukturen der Schnitt führt. Die zuvor eingestellte Schnitttrajektorie und -normale können damit neu beurteilt und gegebenenfalls korrigiert werden. Es handelt sich dabei nur um eine Abschätzung des Abtragsvolumens, die durch die Strahlbreite definierte Schichtdicke dieses Schnitts wird hierbei nicht berücksichtigt. Das später in der Simulation entstehende Abtragsvolumen kann, je nach Lasermodell und abhängig von den realen Prozessparametern (reale Robotertrajektorie, Anzahl der Überfahrungen, Strahlbreite) davon abweichen. Das Volumen wird erzeugt, indem an jedem Punkt der Trajektorie die Raytracingmethode getBeamIntersectingPoints() auf den Octree des Laserobjekts angewandt wird und die gefundenen Voxel zur Visualisierung in das Geoserver-Voxelobjekt geschrieben werden. Die Basis und Richtung des Strahls entsprechen dabei dem jeweiligen Trajektorienpunkt und der Schnittnormalen.

4.5 Modellierung des Laserabtrags

Um den Knochenabtrag berechnen und simulieren zu können, muss dieser zuerst modelliert werden. Dazu ist es nötig, den physikalischen Prozess des Hartgewebeabtrags zu verstehen und mithilfe gewisser Randbedingungen und Vereinfachungen mathematisch zu beschreiben. Mithilfe dieser mathematischen Beschreibung muss dann eine Möglichkeit der Implementierung gefunden werden.

4.5.1 Physikalische Vorgänge

Grundlagen über die Eigenschaften des Knochens sowie des Hartgewebeabtrags wurden bereits in Abschnitt 2.1 beschrieben und werden hier nur kurz zusammengefasst. Unter den beschriebenen Möglichkeiten der Interaktion von Laserstrahlung und biologischem Gewebe wird beim Hartgewebeabtrag mit gepulsten Infrarotlasern das explosionsartige Abtragen erzielt. Dazu sind kurze Pulse und steile Pulsflanken nötig. Im Folgenden wird nur von der TEM_{00}-Mode gesprochen. Aufgrund der gaußschen Energieverteilung des Laserstrahls folgen auch die Krater einer gaußschen Tiefenverteilung, was bei mehreren Pulsen auf dieselbe Stelle in einem immer spitzer zulaufenden Krater resultiert, wie in Abbildung 4.17 zu sehen ist.

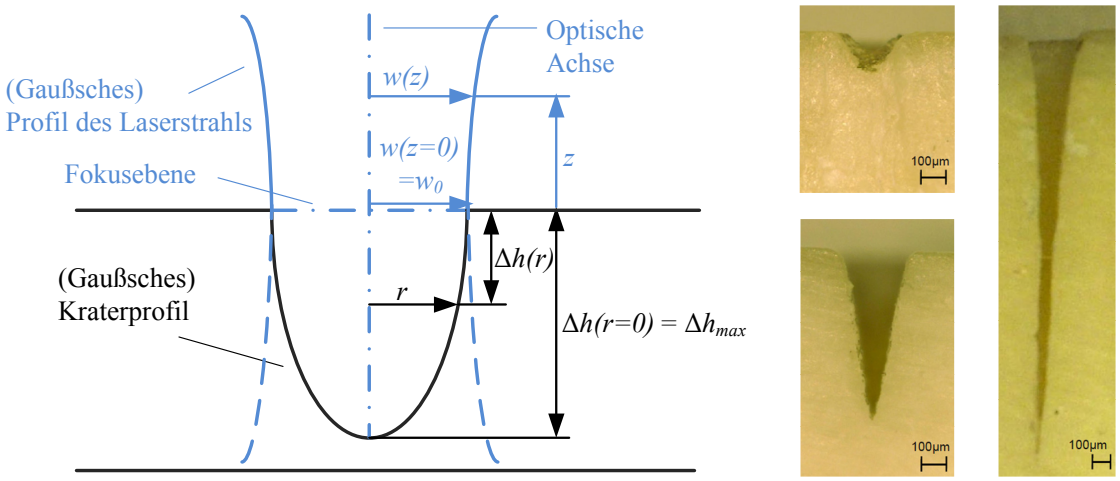


Abbildung 4.17: Links: Gaußscher Strahl und gaußsches Kraterprofil. Rechts: Querschnitt eines Kraters nach 1, 10, 100 Pulsen, entnommen aus [7].

Die maximale Tiefe $\Delta h_{max}\ [mm]$ eines Pulsabtrags (im Mittelpunkt des Strahls) bei optimaler Fokussierung $z = 0$ und senkrechtem Einfall des Laserstrahls auf die Knochenoberfläche hängt dabei von dem Absorptionskoeffizienten $\alpha\ [cm^{-1}]$, der Abtragsschwelle $\Phi_S\ [J/cm^2]$ des Knochens und den Laserparametern Pulsenergie $P_E\ [mJ]$ und Fokusstrahldurchmesser $w_0\ [mm]$ ab. Außerhalb der optischen Achse nimmt die Tiefe $\Delta h(r)$ ab. Auch bei optimaler Nachfokussierung sinkt diese Tiefe mit wachsender Kratertiefe, d.h. beim Anstieg der Pulse auf eine Stelle.

Dies hängt u.a. mit folgenden Phänomenen zusammen:

- Verschmutzen der Linse durch Knochenstaub
- Ansammlung von Wasser im Krater (bei Verwendung von Kühlwasserspray, weniger relevant bei Linienschnitten, da Wasser ablaufen kann)
- Erhöhter Energieabfall an kegelförmigen Flanken, deren Fläche bei wachsender Kratertiefe wächst
- Schwächung der einfallenden Strahlung an der Debris

Diese Phänomene können in einem Abschwächungskoeffizient $\mu\ [cm^{-1}]$ zusammengefasst werden [38], welcher experimentell ermittelt werden muss. Diese Tiefenentwicklung verhält sich logarithmisch, weshalb die Schnitttiefe begrenzt ist. Stopp [56] ermittelte in Experimenten mit Rinderfemur und einem Er:YAG Laser eine nahezu gleichbleibende Schnitttiefe von ca. $1,6\ mm$ ab einer Anzahl von 150 Pulsen.
Zum Erzeugen von Schnitten müssen Krater aneinandergereiht werden. Dazu sind u.a. zwei Strategien (vgl. Abbildung 4.18) möglich:

- Linienschnittstrategie: Lineare Aneinanderreihung der Pulse mit konstanter Pulsüberlappung.
- Wobblestrategie: Überlagerung der Linienbewegung mit einer Kreisbewegung.

Beide Strategien wurden von Burgner für den Hartgewebeabtrag untersucht [7]. Die Wobblestrategie resultierte dabei in aufgeweiteten, stärker kegelförmigen Schnittprofilen. Der Vorteil dessen ist, dass der Effekt der Tiefenlimitierung durch den Abschwächungskoeffizienten $\mu\ [cm^{-1}]$ vermindert wird. Für die Wobblestrategie wird meist ein Scankopf verwendet. In dieser Arbeit wird der Laser direkt vom Roboter geführt und es soll eine möglichst geringe Schnittbreite erzielt werden. Daher wird hier die Linienschnittstrategie verwendet.

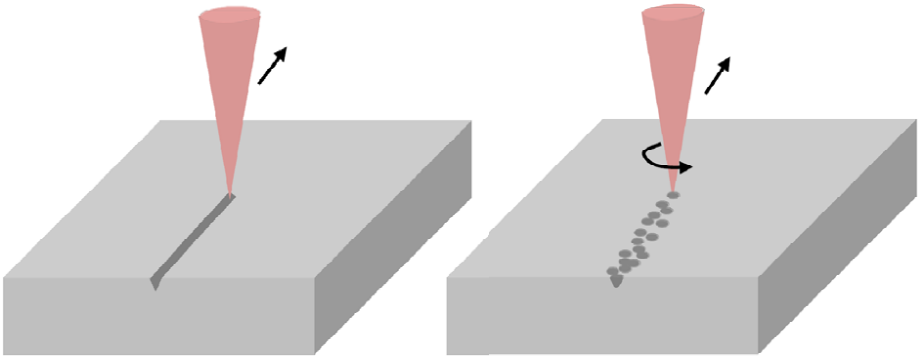


Abbildung 4.18: Verschiedene Strategien zum Lasern eines Schnitts, entnommen aus [7]. Links: Linienschnittstrategie. Rechts: Wobblestrategie.

4.5.2 Mathematisches Modell

Hibst [25] stellte zuerst ein analytisches Modell zur mathematischen Berechnung von Abträgen durch Einzelpulse in Hartgewebe auf. Dieses wurde von Mitra [38] teilweise übernommen und durch Simulationen und Experimente evaluiert. Mit dem Modell kann in Abhängigkeit der Laser- und Materialparameter die maximale Schnitttiefe eines Pulses Δh_{max} berechnet werden. Dabei wird zwischen einem Modell ohne und einem Modell mit Berücksichtigung der Wärmeleitung unterschieden. Bei dem Modell ohne Wärmeleitung wird davon ausgegangen, dass die Erwärmung des Knochens vor Beginn des Abtrags ohne Verluste geschieht. Wie schnell nach Einbringen der thermischen Energie der Abtrag beginnt, hängt von der Abtragsschwelle Φ_S $[J/cm^2]$ des Knochens sowie vom zeitlichen Pulsprofil ab. Bei nahezu rechteckigen Pulsprofilen mit steilen Pulsflanken ist Φ_S schneller erreicht und der Abtrag beginnt früher als bei flachen Pulsflanken. Wie schnell Wärmeverluste durch Wärmeleitung im Material entstehen, kann durch die thermische Relaxationszeit abgeschätzt werden [38]. Die thermische Relaxationszeit ist das Maß, wie schnell die an einen bestimmten Ort eines Materials eingebrachte Wärmeenergie wieder abnimmt. Nach Mitra muss bei CO_2 Lasern die Laserpulsdauer t_p etwa um den Faktor 20 kürzer sein als die thermische Relaxationszeit t_{th}, damit die Wärmeleitung vernachlässigt werden kann. Dies entspricht bei Knochenmaterial und einem CO_2 Laser ($t_{th,CO_2} \approx 20\ \mu s$) ungefähr eine Pulsdauer von $1\ \mu s$. Seine Simulationsergebnisse zeigen außerdem, dass bei höheren Energiedichten ab $40\ J/cm^2$ Wärmeverluste erst ab einer höherer Pulsdauer eintreten. Diesbezüglich ist ein zeitliches Pulsprofil, welches über die gesamte Pulsdauer annähernd die gleiche Energie in das Material einbringt, günstig. Der hier verwendete Er:YAG Laser AT Fidelis mit einer minimalen Pulsdauer von $50\ \mu s$ erfüllt die erwähnte maximale Pulsdauer von $1\ \mu s$ nicht. Die maximalen Energiedichte des AT Fidelis von $48\ J/cm^2$ liegt zwar über den geforderten $40\ J/cm^2$ um auch bei längerer Pulsdauer ohne Wärmeleitung rechnen zu können, jedoch werden von Mitra hierzu keine quantitativen Angaben gemacht, wie lang die Pulsdauer in diesem Fall sein darf. Er führte seine Experimente mit einem CO_2 Laser durch, bei dessen Wellenlänge ($\lambda_{CO_2} \approx 10\mu m$) Knochen einen niedrigeren Absorptionskoeffizienten α und damit eine höhere Eindringtiefe hat als bei dem hier verwendeten Er:YAG Laser ($\lambda_{ER:YAG} \approx 2.9\mu m$). Die thermische Relaxationszeit ist umgekehrt proportional zur Eindringtiefe, weshalb sie beim Er:YAG Laser höher liegt als beim CO_2 Laser. Der Er:YAG Laser AT Fidelis zeichnet sich außerdem durch steile Pulsflanken aus, weshalb die Abtragsschwelle $\Phi_S[J/cm^2]$ des Knochens schnell erreicht ist und der Abtrag früh beginnt. Aus diesen Gründen soll in vorliegender Arbeit das Modell ohne Berücksichtigung der Wärmeleitung verwendet werden. Bei diesem ist die maximale Kratertiefe Δh_{max} bei optimaler Fokussierung des Strahls ($z = 0$) und in der Mitte des Kraters ($r = 0$) mit den Variablen Absorptionskoeffizient α, Abschwächungskoeffizient μ, Abtragsschwelle Φ_S, sowie der maximalen Energiedichte Φ_{max} wie folgt bestimmt:

$$\Delta h_{max} = \frac{1}{\mu} \cdot \ln\left(\frac{\mu}{\alpha} \cdot \left(\frac{\Phi_{max}}{\Phi_S} - 1\right) + 1\right) \tag{4.25}$$

Bei Abweichung vom Fokus ($z \neq 0$) sinkt die Energiedichte. Außerdem ist der Energieeintrag über den Strahlenquerschnitt nicht konstant, andernfalls würde ein zylinderformiger Krater entstehen. Stattdessen folgt sie einer gaußschen Energieverteilung, so dass die Gleichung 4.25 verallgemeinert und die Schnitttiefe $\Delta h(z,r)$ an jeder Stelle der bestrahlten Fläche berechnet werden kann.

$$\Delta h(z,r) = \frac{1}{\mu} \cdot \ln\left(\frac{\mu}{\alpha} \cdot \left(\frac{\Phi(z,r)}{\Phi_S} - 1\right) + 1\right) \tag{4.26}$$

Die vom Radius- und vom Fokusabstand abhängige Energiedichte $\Phi(z,r)$ berechnet sich mit der Pulsenergie P_E, dem aktuellen Strahlradius $w(z)$, sowie dem Abstand von der optischen Achse r:

$$\Phi(z,r) = \frac{2 \cdot P_E}{\pi \cdot w(z)^2} \cdot e^{-2\frac{r^2}{w(z)^2}} \tag{4.27}$$

Der aktuelle Strahlradius lässt sich mit der Rayleighlänge z_r und dem Abstand vom Fokus z berechnen, so dass gilt:

$$w(z) = w_0 \cdot \sqrt{1 + \left(\frac{z}{z_r}\right)^2} \tag{4.28}$$

Mit den Gleichungen 4.25 bis 4.28 ist ein Einzelpulsabtrag analytisch beschreibbar. In diesem Modell wird eine Abweichung von der Fokusebene sowie eine gaußsche Energieverteilung über den Strahlenquerschnitt berücksichtigt, nicht aber eine Abweichung vom Einfallswinkel. Diese Abweichung wird bei einem Modell von Burgner [7] berücksichtigt. Sie wird hier vernachlässigt, da davon ausgegangen wird, dass der Roboter den Laserstrahl immer senkrecht zur Knochenoberfläche ausrichten kann. Auch die Abhängigkeit des Strahlradius $w(z)$ vom Abstand zur Fokusebene z wird vernachlässigt, da davon ausgegangen wird, dass der Roboter den Laserstrahl immer optimal fokussiert und der Strahlradius immer w_0 beträgt. Die logarithmische Tiefenentwicklung, d.h. die sinkende Schnittgeschwindigkeit bei mehreren Pulsen auf eine Stelle und steigender Kratertiefe h, wird im Abschwächungskoeffizient μ modelliert, so dass dieser zu $\mu(h)$ erweitert wird. Dadurch entfällt Gleichung 4.28 und die Gleichungen 4.26 sowie 4.27 vereinfachen sich zu folgenden zwei Gleichungen 4.29 und 4.30:

$$\Delta h(r,h) = \frac{1}{\mu(h)} \cdot \ln\left(\frac{\mu(h)}{\alpha} \cdot \left(\frac{\Phi(r)}{\Phi_S} - 1\right) + 1\right) \tag{4.29}$$

$$\Phi(r) = \frac{2 \cdot P_E}{\pi \cdot w_0^2} \cdot e^{-2\frac{r^2}{w_0^2}} \tag{4.30}$$

Zur Ermittlung von $\mu(h)$ wurden Experimente aus der Literatur betrachtet, welche zur Tiefenentwicklung durchgeführt wurden. Stopp [56] führte Experimente durch, bei denen mit einem Er:YAG Laser unterschiedliche Anzahlen von Pulsen $N \in \{10; 20; ...; 150\}$ in die Kompakta eines Schweinefemurs eingebracht wurden und die Entwicklung der Kratertiefe $h(N)$ betrachtet wurde. Das Experiment ist sehr ausführlich und gut dokumentiert, jedoch wird ein anderer Laser (Kavo Key Laser 3) verwendet was zur Folge hatte, dass ein großer Anteil der dabei ermittelten Abschwächung auf die wachsende Wasseransammlung im tiefer werdenden Krater zurückzuführen sei. Bei einem Linienschnitt verringert sich dieser Effekt unter Umständen, da das Wasser ablaufen kann. Deshalb wurde ein Experiment gesucht, welches die Tiefenentwicklung bei Linienschnitten mit dem Laser AT Fidelis betrachtet. Im Rahmen zweier Bachelorarbeiten

[4], [19] am DLR wurde ein solches Experiment durchgeführt. Dabei wurden mit folgenden Parametern Linien in die Kompakta eines Schweinefemurs gelasert.

- 2, 4, 16 Überfahrungen
- Konstante absolute Geschwindigkeit: $|\boldsymbol{v}| = 10\ mm/s$
- Frequenz: $f_p = 20\ Hz$
- Fokusabstand: $z = 0$
- Einfallswinkel des Laserstrahls: $\phi = 0$
- Pulsenergie: $P_E = 0,75\ J$

Aus der Geschwindigkeit und der Frequenz lässt sich die Pulsüberlappung Δp berechnen. Sie beträgt $0,5\ mm$ und entspricht damit annähernd dem Strahlradius des AT Fidelis $r_0 = 0,55\ mm$ [4]. Bei Vereinfachung der gaußschen Kraterform zu einer kegelförmigen Form (vgl. Abbildung 4.19) kann angenommen werden, dass so eine Linie entsteht, deren Tiefe der eines Einzelkraters entspricht.

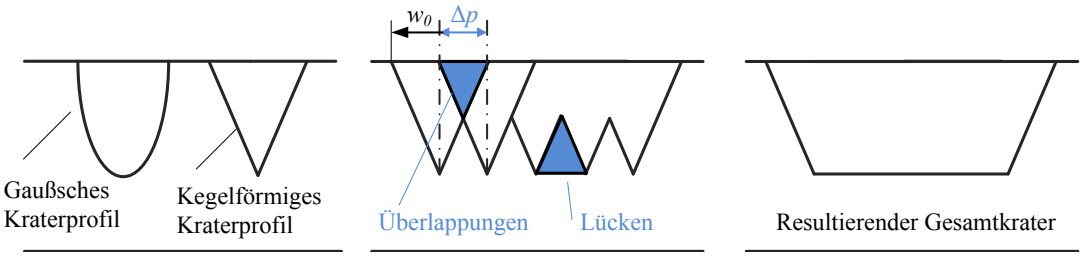


Abbildung 4.19: Links: Vereinfachung des gaußschen Kraters zu einem kegelförmigen Krater. Mitte: Fünf aneinandergereihte kegelförmige Krater mit konstanter Pulsüberlappung Δp. Rechts: Daraus resultierender Gesamtkrater.

Die Linientiefe nach N Überfahrungen entspricht unter diesen Annahmen einer mit Gleichung 4.29 zu berechnenden Kratertiefe nach N Einzelpulsen. Mit den Ergebnissen des Experiments wurde eine Fitting Curve gebildet, welche in Abbildung 4.20 auf der linken Seite zu sehen ist. Dort wird $h(N)$ der Anzahl der Pulse N gegenübergestellt, der Verlauf ist logarithmisch.

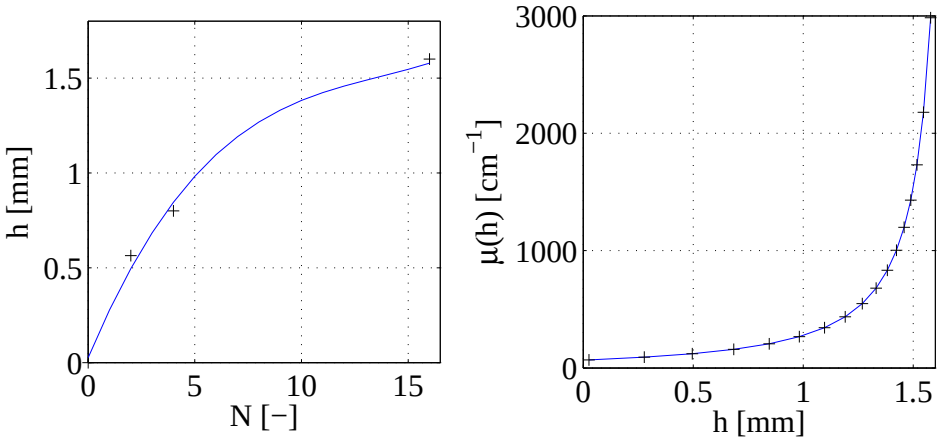


Abbildung 4.20: Links: Fitting Curve über die Ergebnisse des Experimentes zur Tiefenentwicklung h abhängig von der Anzahl der Pulse N. Rechts: Experimentell ermittelter Abschwächungskoeffizient $\mu(h)$ abhängig von der Kratertiefe h.

Das Differenzial der Kratertiefe $h(N)$ entspricht der Schnitttiefe $\Delta h(N)$ nach N eingebrachten Pulsen. Bei Gegenüberstellung von $\Delta h(N)$ und $h(N)$, kann $\Delta h(h)$ gebildet und mit folgendem Polynom (gilt nur für den hier angegebenen Bereich) angenähert werden:

$$\Delta h(h) = -0.0040 \cdot h^2 - 0.1465 \cdot h + 0.2590 \tag{4.31}$$

Die Gleichungen 4.29 (mit $r = 0$) und 4.31 können gleichgesetzt und der Abschwächungskoeffizient $\mu(h)$ daraus errechnet werden.

$$\frac{1}{\mu(h)} \cdot \ln\left(\frac{\mu(h)}{\alpha} \cdot \left(\frac{\Phi_{max}}{\Phi_S} - 1\right) + 1\right) = 0.0093 \cdot h^2 - 0.0453 \cdot h + 0.0488 \tag{4.32}$$

Da eine spätere Verarbeitung der Werte im Algorithmus diskret geschieht, wurde Gleichung 4.32 nicht analytisch nach $\mu(h)$ aufgelöst, sondern es wurde alle $0,1\ mm$ von h ein Wert für $\mu(h)$ gebildet. So konnte das Problem als Nullstellenproblem in Matlab gelöst werden, in einem Bereich von $h \in \{0; 1; ...; 1,6\}$ ergaben sich dadurch 16 Werte, welche in Abbildung 4.20 auf der rechten Seite zu sehen sind. Die restlichen dazu benötigten Parameter wurden aus der Literatur entnommen:

- Absorptionskoeffizient von Kompakta bei $\lambda = 2,94\ \mu m$: $\alpha = 3800\ cm^{-1}$ [38]
- Absorptionsschwelle von Kompakta: $\Phi_S = 0,32\ J/cm^2$ [56]
- Maximale Energiedichte berechnet mit $w_0 = 0,55\ mm$ [4] und Gleichung 4.30: $\Phi_{max} = 84,9\ J/cm^2$

Die Ergebnisse von $\mu(h)$ decken sich mit den Annahmen von Mitra, dass der Abschwächungskoeffizient zwischen den beiden Grenzfällen $\mu = 0$ und $\mu = \alpha$ liegen muss [38]. Mit den ermittelten und aus der Literatur übernommenen Parametern sowie den getroffenen Idealisierungen kann mithilfe von Gleichung 4.29 nun an jeder bestrahlten Stelle die aus einem Puls resultierende Schnitttiefe $\Delta h(r, h)$ berechnet werden.

4.5.3 Implementierung des Modells

Zur Implementierung muss das mathematische Modell mit der Volumenmodellierung in Einklang gebracht werden. Wie in Abschnitt 4.4 erklärt, ist das Volumen in einem Octree abgespeichert, wobei mit einer gewissen Diskretisierung jedem Punkt im Raum die Information über einen Gesundheitszustand ($H \in [0; 100]$) zugeordnet ist. Daher muss auch das mathematische Modell des Laserabtrags diskretisiert werden, so dass der Energieeintrag in einzelne Voxel eingebracht und deren Gesundheitszustand entsprechend verringert wird. Die getroffenen Annahmen und Vereinfachungen sind folgende:

- Der gaußsche Laserstrahl mit variablem Strahlradius wird zu eine zylindrischen Strahl mit konstantem Strahlradius w_0 vereinfacht
- Die Energiedichte $\Phi(r)$ ist nur vom Radius r, nicht vom Fokusabstand z abhängig (vgl. Gleichung 4.30)
- Die Schnitttiefe eines Einzelpulses $\Delta h(r, h)$ ist vom Radius r und von der wachsenden Kratertiefe h abhängig (vgl. Gleichung 4.29)

Im Rahmen dieser Arbeit wurden zwei Einzelpulsmodelle M_M implementiert, eines berechnet einen gaußschen Krater und eines berechnet einen vereinfachten zylindrischen Krater. Beide werden im Folgenden erklärt.

Modell Gaußscher Krater Das Einzelpulsmodell Gaußscher Krater M_G ist an ein von Stopp [56] beschriebenes Modell angelehnt. Es kann in vier Einzelschritte unterteilt werden.

- Schritt 1: Finden des Auftreffpunkts
- Schritt 2: Bilden eines Zylinders
- Schritt 3: Finden der im Zylinder liegenden Voxel und Sortieren dieser nach Tiefe
- Schritt 4: Berechnen der Verminderung des Gesundheitszustands H dieser Voxel

Zuerst muss in Schritt 1 mittels der Raytracingfunktion (vgl. Abschnitt 4.4) getBeamIntersectingPoints() ein Auftreffpunkt der optischen Achse des Laserstrahls auf das Voxelvolumen des Laserobjekts gefunden werden. Als Auftreffpunkte werden nur solche Voxel akzeptiert, für deren Gesundheitszustand $H > 0$ gilt, d.h. solche, die noch nicht vollständig abgetragen sind. Liegt H bei 100, so wird als Auftreffpunkt die Voxeloberfläche verwendet. Liegt H unter 100, so bedeutet dies, dass das Voxel durch einem vorherigen Puls

zum Teil abgetragen wurde. Der Auftreffpunkt wird in diesem Fall um folgenden Betrag Δa nach unten verschoben. Die Variable $voxsize$ entspricht der Kantenlänge eines Voxels.

$$\Delta a = \frac{H}{100} \cdot voxsize \tag{4.33}$$

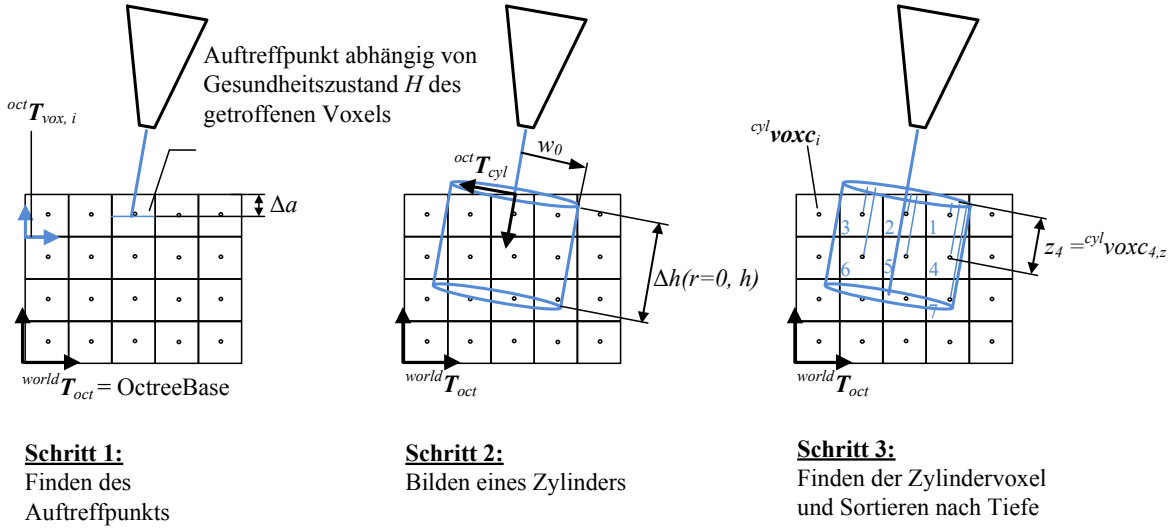


Abbildung 4.21: Erste drei Schritte des Einzelpulsmodells M_G.

In Schritt 2 wird parallel zur optischen Achse ein Zylinder gebildet, dessen Basis ${}^{oct}\boldsymbol{T}_{cyl}$ im Auftreffpunkt liegt. Der Zylinderradius richtet sich nach dem Strahldurchmesser w_0, die Schnitttiefe in der optischen Achse $\Delta h_{max}(r, h) = \Delta h(r = 0, h)$ (vgl. Gleichung 4.29) legt die Tiefe des Zylinders fest. Im Allgemeinen wird bei dieser Zylindergröße von der Zylinderhöhe gesprochen, da es sich hierbei aber um die Schnitttiefe handelt, soll sie jedoch im Folgenden als Tiefe bezeichnet werden. Die in diesem Zylinder liegenden Voxel werden in Schritt 3 in einem Container gesammelt und nach ihrer Tiefe, d.h. dem Abstand von der oberen Fläche des Zylinders geordnet. Die Basis in Frage kommender Voxel i werden dafür von Octreekoordinaten ${}^{oct}\boldsymbol{T}_{vox_i}$ in das Koordinatensystem der Zylinderbasis ${}^{cyl}\boldsymbol{T}_{vox_i}$ umgerechnet. Dazu sei gesagt, dass im Folgenden die Konvention verwendet wird einen Punkt $\boldsymbol{p}$ (hier die Basis des Voxels i) im Koordinatensystem K ($\boldsymbol{T}_K$) je nach Anforderung als homogene Matrix ${}^K\boldsymbol{T}_p$ oder als Vektor ${}^K\boldsymbol{p}$, welcher den translatorischen Anteil der homogenen Matrix beschreibt, zu verwenden. Mit einer homogenen Matrix können Translationen und Rotationen, wie sie bei einer Koordinatentransformation vorkommen in einem Schritt als Matrixprodukt vollzogen werden (vgl. Gleichung 4.37). Der rotatorische Anteil der homogenen Matrix eines Punkts ohne definierte Orientierung entspricht der Einheitsmatrix.

$${}^K\boldsymbol{T}_p = \begin{bmatrix} 1 & 0 & 0 & {}^Kp_x \\ 0 & 1 & 0 & {}^Kp_y \\ 0 & 0 & 1 & {}^Kp_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.34}$$

$$ {}^{K}\boldsymbol{p} = \begin{bmatrix} {}^{K}p_x \\ {}^{K}p_y \\ {}^{K}p_z \end{bmatrix} \tag{4.35} $$

Zur Umrechung des Voxels i von Octreekoordinaten nach Koordinaten der Zylinderbasis muss vorerst die Transformationsmatrix ${}^{cyl}\boldsymbol{T}_{oct}$ von Octree- nach Zylinderkoordinaten gebildet werden:

$$ {}^{cyl}\boldsymbol{T}_{oct} = ({}^{oct}\boldsymbol{T}_{cyl})^{-1} \tag{4.36} $$

$$ {}^{cyl}\boldsymbol{T}_{vox_i} = {}^{cyl}\,\boldsymbol{T}_{oct} \cdot {}^{oct}\,\boldsymbol{T}_{vox_i} \tag{4.37} $$

Das Koordinatensystem der Zylinderbasis ${}^{oct}\boldsymbol{T}_{cyl}$ ist ein kartesisches Koordinatensystem mit x-, y- und z-Achse. Die z-Achse entspricht der Strahlrichtung, der Betrag $(x^2 + y^2)$ entspricht der radialen Ausrichtung im Zylinder. Da die Voxelbasis ${}^{cyl}\boldsymbol{T}_{vox_i}$ bzw. ${}^{cyl}\boldsymbol{vox}_i$ in der Ecke des Voxels liegt, eignet sie sich nicht um zu bestimmen, ob ein Voxel innerhalb oder außerhalb des Zylinders liegt. Daher muss zuerst der Voxelmittelpunkt ${}^{cyl}\boldsymbol{T}_{voxc_i}$ bzw. ${}^{cyl}\boldsymbol{voxc}_i$ gebildet werden.

$$ {}^{cyl}\boldsymbol{voxc}_i = \begin{bmatrix} {}^{cyl}vox_{i,x} + voxsize/2 \\ {}^{cyl}vox_{i,y} + voxsize/2 \\ {}^{cyl}vox_{i,z} + voxsize/2 \end{bmatrix} \tag{4.38} $$

Folgende zwei Bedingungen beschreiben, ob ein Voxelmittelpunkt ${}^{cyl}\boldsymbol{voxc}_i$ im Zylinder liegt. Damit wird gemäß Schritt 3 in Abbildung 4.21 ein Voxel dem Zylinder $\mathbb{Z}$ als zugehörig definiert ($Voxel \in \mathbb{Z}$):

$$ {}^{cyl}voxc_{i,z} < \Delta h_{max}(r,h) + voxsize/2 \tag{4.39} $$

$$ ({}^{cyl}voxc_{i,x}^2 + {}^{cyl}\,voxc_{i,y}^2) < w_0 \tag{4.40} $$

Das in Gleichung 4.39 addierte $voxsize/2$ ermöglicht, dass auch Voxel, deren Mittelpunkt ${}^{cyl}\boldsymbol{voxc}_i$ maximal $voxsize/2$ unterhalb des Zylinders liegen, noch als zum Zylinder zugehörig definiert werden. Radial werden nur Voxel, deren Mittelpunkt innerhalb des Zylinders liegen, als ihm zugehörig definiert. Da solche geometrischen Operationen rechenintensiv sind, sollen nicht alle Voxel im Octree zuerst mit Gleichung 4.37 transformiert werden um dann mit den Bedingungen aus Gleichung 4.39 und 4.40 auf Zugehörigkeit zum Zylinder geprüft zu werden. Stattdessen wird eine Voxelmenge $\mathbb{V}_{cube}$, deren Voxel sich in der Nähe des Zylinders befinden, vorausgewählt. Die C++ Klasse Octree (vgl. Abschnitt 4.4) stellt die einfache und effektive Methode getIntersectingObjects() bereit. Die Methode findet eine Voxelmenge $\mathbb{V}_{cube}$ innerhalb des Octrees $\mathbb{O}$, deren Voxel sich in einem durch Basis und Kantenlänge definierten Kubus $\mathbb{C}$ befinden.

$$ \mathbb{V}_{cube} = \mathbb{O} \cap \mathbb{C} \tag{4.41} $$

Es wird also vor dem eigentlichen Schritt 3, dem Zuordnen der Voxel zum Zylinder, mit getIntersectingObjects() die Voxelmenge $\mathbb{V}_{cube}$ gebildet. Die Effektivität der Methode basiert darauf, dass sie in Voxelkoordinaten und nicht in den für die Transformationen verwendeten absoluten Koordinaten rechnet. Voxel 6 in Abbildung 4.21 wird beispielsweise in Octreekoordinaten durch $(1|2)$ beschrieben, während seine Position in absoluten Koordinaten durch die Position seiner Basis ${}^{oct}\boldsymbol{T}_{vox_6} = (1 \cdot voxsize \mid 2 \cdot voxsize)$ gegeben ist. Daher muss der verwendete Kubus, wie in Abbildung 4.22 zu sehen ist, parallel zum Octree orientiert sein. Eine Kippung parallel zur Zylinderbasis ${}^{oct}\boldsymbol{T}_{cyl}$ ist nicht möglich. Um unter dieser Bedingung einen Kubus zu bilden, der den Zylinder immer vollständig umfasst, wird die Kantenlänge des Kubus $cubesize$ und dessen Basis ${}^{oct}\boldsymbol{T}_{\mathbb{C}}$ wie folgt definiert. Dabei ist ${}^{oct}\boldsymbol{cyl}$ der translatorische Anteil der Zylinderbasis ${}^{oct}\boldsymbol{T}_{cyl}$, die Größe d hängt vom Radius w_0 und der Tiefe $\Delta h(r = 0, h)$ des Zylinders ab. Dies ist in Abbildung 4.22 verdeutlicht.

$$cubesize = 2 \cdot d = 2 \cdot (w_0^2 + \Delta h(r = 0, h)^2) \tag{4.42}$$

$${}^{oct}\boldsymbol{T}_{\mathbb{C}} = \begin{bmatrix} 1 & 0 & 0 & {}^{oct}cyl_x - cubesize/2 \\ 0 & 1 & 0 & {}^{oct}cyl_y - cubesize/2 \\ 0 & 0 & 1 & {}^{oct}cyl_z - cubesize/2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.43}$$

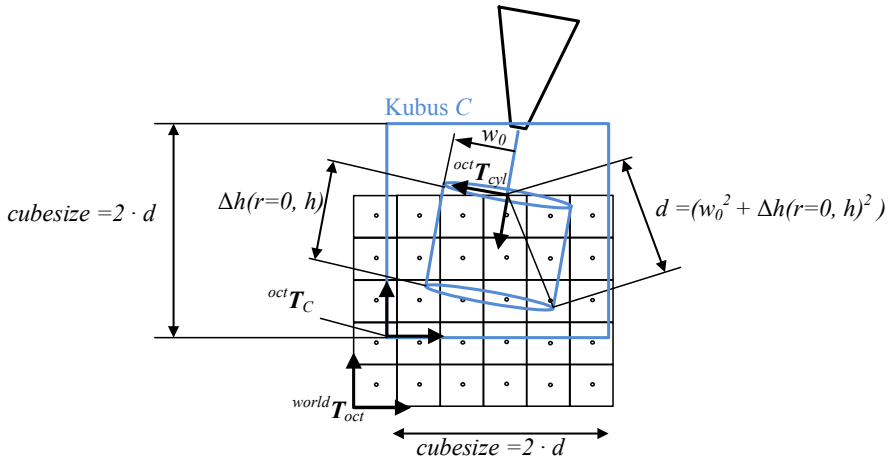


Abbildung 4.22: Zwischenschritt des Modells M_G: Definition eines Kubus, der den Zylinder umschließt.

Die Voxel aus der vorausgewählten Menge $\mathbb{V}_{cube}$ können dann gemäß Gleichung 4.39 und 4.40 auf Zugehörigkeit zum Zylinder geprüft werden, was einen Teil des in Abbildung 4.21 dargestellten Schritt 3 ausmacht. In Schritt 3 werden diese Voxel aber auch nach ihrer Tiefe z_i im Zylinder geordnet. z_i entspricht dem z-Anteil des translatorischen Anteils des Voxelmittelpunkts ${}^{cyl}voxc_{i,z}$. Nun liegt eine neue Menge $\mathbb{V}_{cyl}$ von Voxeln vor, welche im Zylinder liegen und nach z_i geordnet sind. In dieser Menge wird von z_{min} bis z_{max} die Schnitttiefe $\Delta h(r_i, h)$ an der Stelle jedes Voxels i mit $r_i = ({}^{cyl}voxc_{i,x}^2 + {}^{cyl}\ voxc_{i,y}^2)$ und der momentanen Kratertiefe h berechnet. Damit kann die Minderung des Gesundheitszustands H_i der einzelnen Voxel ermittelt werden.

Anhand von Abbildung 4.23 kann der Zusammenhang zwischen Gesundheitszustand H und Schnitttiefe $\Delta h(r, h)$ nachvollzogen werden.

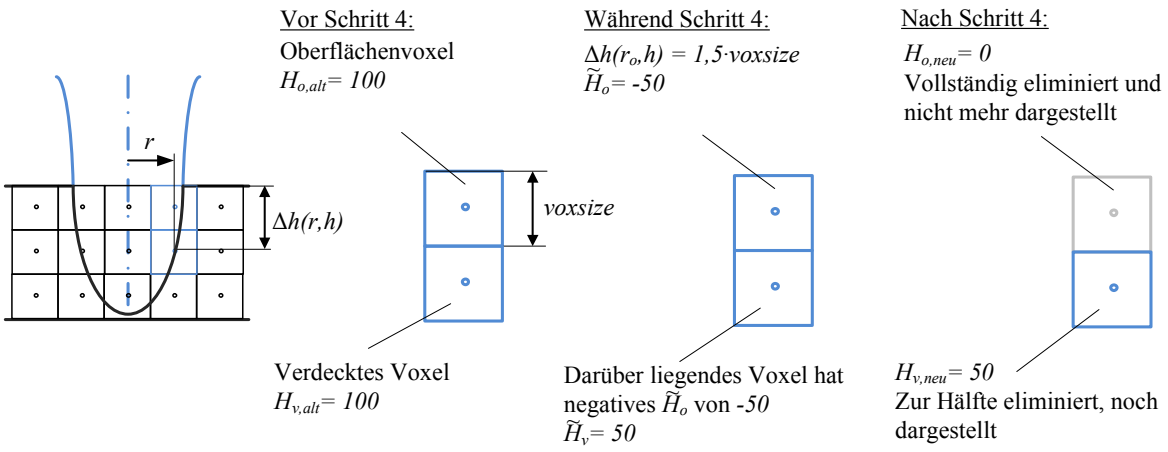


Abbildung 4.23: Zusammenhang zwischen dem Gesundheitszustand H und der Schnitttiefe $\Delta h(r, h)$.

Ein Voxel i mit einem Gesundheitszustand von $H_i = 100$ bedeutet, dass an dieser Stelle des Knochens, definiert durch ${}^{oct}\boldsymbol{voxc}_i$ ein vollwertiges Voxel der Kantenlänge $voxsize$ existiert. Der Gesundheitszustand von $H_i = 50$ impliziert, dass an dieser Stelle des Knochens ein Voxel der Kantenlänge $voxsize/2$ besteht. Bei einem Gesundheitszustand von $H_i = 0$ existiert an dieser Stelle kein Voxel mehr. So ist eine Genauigkeit der Kratertiefe möglich, welche unabhängig von der Auflösung der Diskretisierung ist. Je nach Auflösung ist es möglich, dass die nach Gleichung 4.29 berechnete Schnitttiefe $\Delta h(r, h)$ eines getroffenen Oberflächenvoxels die Voxelkantenlänge $voxsize$ übertrifft. In diesem Fall wird, wie in Abbildung 4.23 zu sehen ist, während der Berechnung eines Einzelpulsabtrags über die Menge $\mathbb{V}_{cyl}$, der Gesundheitszustand dieses Oberflächenvoxels H_o auf einen negativen Wert gesetzt (z.B. $\tilde{H}_o = -50$), um in einem weiteren Berechnungsschritt an das darunterliegende verdeckte Voxel weitergegeben zu werden. Die Gesundheitszustände vor, während und nach dieser Berechnung werden mit H_{alt}, $\tilde{H}$ und H_{neu} bezeichnet. Dabei werden dem Gesundheitswert des darunterliegenden, verdeckten Voxels H_v die 50 abgezogen, so dass am Ende $H_{o,neu} = 0$ und $H_{v,neu} = 50$ resultiert. Bei sehr geringer Auflösung können zwischenzeitlich Gesundheitszustände von $\tilde{H} < -100$ entstehen, so dass $\tilde{H}$ über mehrere Voxel nach unten gegeben werden muss. Bei der Berechnung eines Einzelpulsabtrags über die Gesamtvoxelmenge $\mathbb{V}_{cyl}$ ist daher entscheidend, ob ein Voxel an der Oberfläche liegt und direkt vom Laserstrahl getroffen wird, oder ob es verdeckt ist. Voxel mit niedrigem z liegen an der Oberfläche, Voxel mit hohem z sind verdeckt, daher ist es sinnvoll die Menge $\mathbb{V}_{cyl}$ von z_{min} bis z_{max} durchzuarbeiten um die Schnitttiefe $\Delta h(r, h)$ und den daraus resultierenden Gesundheitszustand H der einzelnen Voxel zu berechnen. Dies geschieht gemäß Abbildung 4.24.

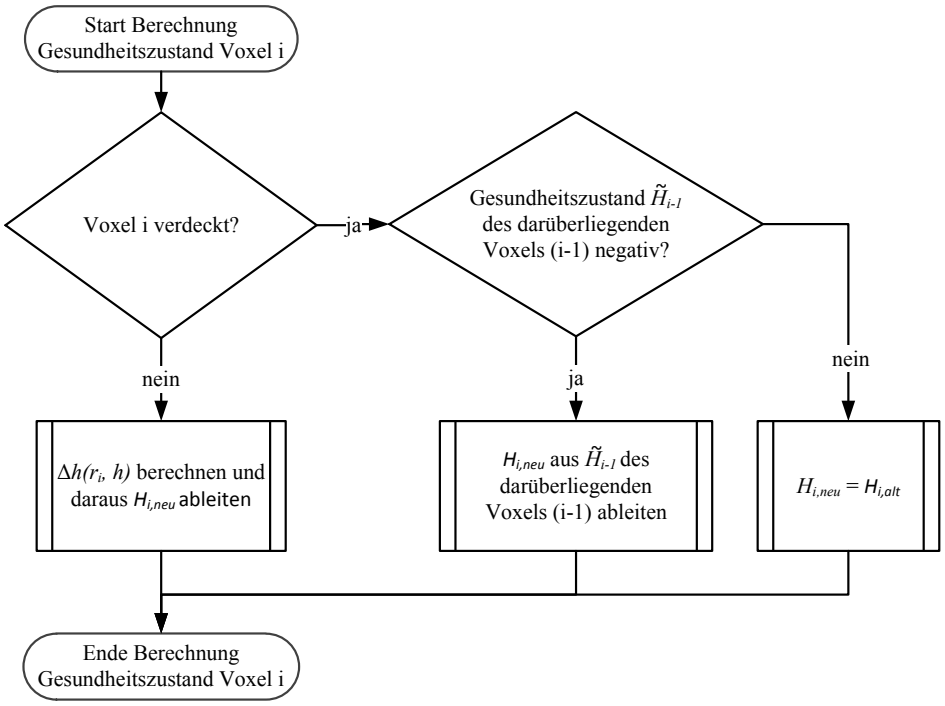


Abbildung 4.24: Flussidagramm zur Berechnung des neuen Gesundheitszustands H_{neu} der Voxel in $\mathbb{V}_{cyl}$ (Schritt 4 des Modells).

Wie in der Abbildung zu sehen ist, muss in Schritt 4 für alle Voxel i aus $\mathbb{V}_{cyl}$ die Schnitttiefe $\Delta h(r_i, h)$ nach Gleichung 4.29 berechnet werden, wenn sie Oberflächenvoxel sind. Dies geschieht mit dem Radius des Voxels i ($r_i =^{cyl} voxc^2_{i,x} +^{cyl} voxc^2_{i,y}$) und der momentanen Kratertiefe h. Der während der Berechnung vorübergehende Gesundheitszustand $\tilde{H}_i$ wird aus dem alten Energiezustand $H_{i,alt}$ und $\Delta h(r_i, h)$ wie folgt berechnet:

$$\tilde{H}_i = H_{i,alt} - \frac{\Delta h(r_i, h)}{voxsize} \tag{4.44}$$

Ist Voxel i verdeckt, so wird sein Gesundheitszustand vom Gesundheitszustand $\tilde{H}_{i-1}$ des darüberliegenden Voxels (i-1) abgeleitet. Durch die Berechnungsreihenfolge von $\mathbb{V}_{cyl}$ von z_{min} nach z_{max} ist garantiert, dass der Gesundheitszustand $\tilde{H}_{i-1}$ zu diesem Zeitpunkt schon berechnet wurde.

$$\tilde{H}_i = \begin{cases} H_{i,alt} & \text{wenn } \tilde{H}_{i-1} \geq 0 \\ H_{i,alt} + \tilde{H}_{i-1} & \text{wenn } \tilde{H}_{i-1} < 0 \end{cases} \tag{4.45}$$

Am Ende der Berechung werden alle Gesundheitszustände $\tilde{H}_i$ auf Null gesetzt, sofern sie negativ sind, und übernommen, sofern sie positiv sind. Negative Energiezustände bestehen damit nur während einer solchen Berechnung.

$$H_{i,neu} = \begin{cases} \tilde{H}_i & \text{wenn } \tilde{H}_i > 0 \\ 0 & \text{wenn } \tilde{H}_i \leq 0 \end{cases} \tag{4.46}$$

Ob ein Voxel i ein Oberflächenvoxel ist, wird mithilfe der Raytracingfunktion getBeamIntersectingPoints() festgestellt. Dabei wird ein Strahl gebildet, dessen Basis dem Mittelpunkt des Voxels i ${}^{cyl}\boldsymbol{voxc}_i$ und dessen Richtung der umgekehrten Richtung des Laserstrahls entspricht. Sofern das Voxel i ein verdecktes Voxel ist, werden die darüber liegenden Voxel zwangsläufig von diesem Strahl getroffen. Ist das Voxel i ein Oberflächenvoxel, so werden vor ihm keine anderen Voxel getroffen.

Modell Gaußscher Krater bei mehreren Pulsen Wie beschrieben, richtet sich die Basis des in Schritt 2 (vgl. Abbildung 4.21) zu bildenden Zylinders beim ersten Puls nach dem Auftreffpunkt, die Tiefe richtet sich nach der Schnitttiefe in der optischen Achse $\Delta h(r = 0, h)$. Wird dies beim zweiten Puls auf dieselbe Weise getan, so tritt der in Abbildung 4.25 dargestellte Fehler auf. Statt dass der Krater beim Aufbringen mehrerer Pulse die charakteristische Kegelform annimmt, entwickelt sich ein gezackter Krater. Daher muss die Basis ab dem ersten Puls gleich bleiben, während die Tiefe des Zylinders dep_{cyl} sich aus folgender Summe ergibt:

$$dep_{cyl} = \Delta h(r = 0, h) + \Delta AP \tag{4.47}$$

ΔAP gibt dabei die Differenz des ersten und des aktuellen Auftreffpunkts der Raytracingfunktion an. Daraus ergibt sich, dass der Zylinder $\mathbb{Z}$ und damit die in Schritt 4 zu berechnende Voxelmenge $\mathbb{V}_{cyl}$ wächst.

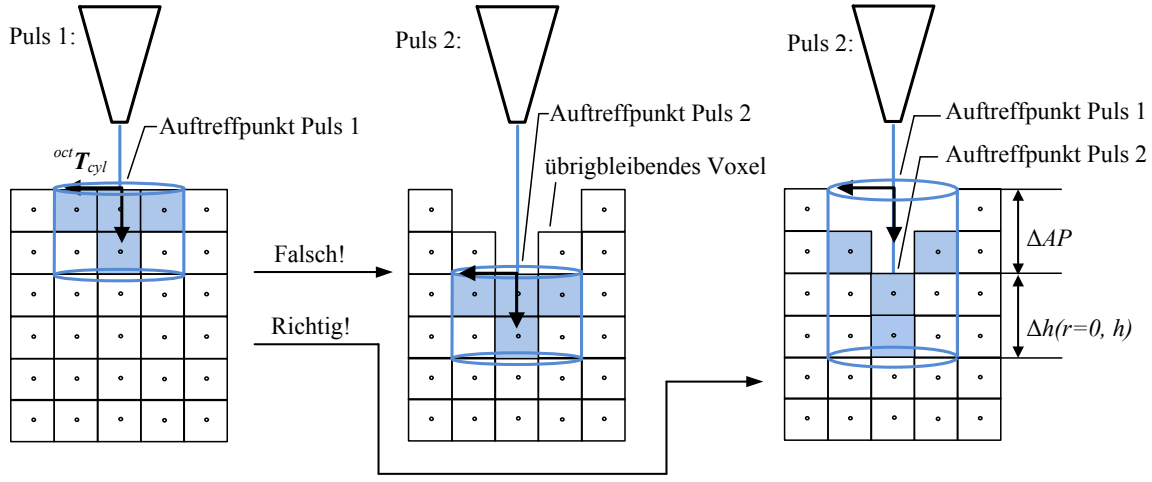


Abbildung 4.25: Problem des Modells M_G: Bei mehreren Pulsen wächst der Zylinder, in welchem die Voxel geprüft werden müssen.

Modell Zylindrischer Krater Aufgrund hoher Rechenzeiten des Modells M_G wurde das vereinfachte Modell Zylindrischer Krater M_Z implementiert, welches den gaußschen Krater zu einem zylindrischem Krater reduziert. Damit können mehrere Rechenschritte vermieden werden. Zum einen wird das im vorherigen Paragraph beschriebene Problem vermieden, dass die Voxelmenge $\mathbb{V}_{cyl}$ bei steigender Pulsanzahl wächst. Das Modell M_Z kann wie das Modell M_G in vier Einzelschritte unterteilt werden, wobei sich die Schritte 1-3 kaum unterscheiden. Auch hier wird im Schritt 1 mit der Raytracingfunktion getBeamIntersectingCubes() ein Auftreffpunkt gesucht, wenn für den Gesundheitszustand des Voxels i des Auftreffpunkts $H_i < 100$ gilt, so wird der Auftreffpunkt entsprechend Gleichung 4.33 um Δa verschoben. In Schritt 2 wird ein Zylinder gebildet, dessen Basis ${}^{oct}\boldsymbol{T}_{cyl}$ im Auftreffpunkt liegt. Der Zylinderradius entspricht w_0 und seine Tiefe entspricht $\Delta h(r = 0, h)$. Im Gegensatz zum Modell Gaußscher Krater wird die Basis ${}^{oct}\boldsymbol{T}_{cyl}$ nicht nur beim ersten sondern auch beim i-ten Puls im Auftreffpunkt gebildet, wie in Abbildung 4.26 zu sehen ist. Das im vorherigen Paragraph beschriebene Problem besteht dabei nicht, da die Kraterwände senkrecht abfallen und so keine Voxel übrig bleiben können.

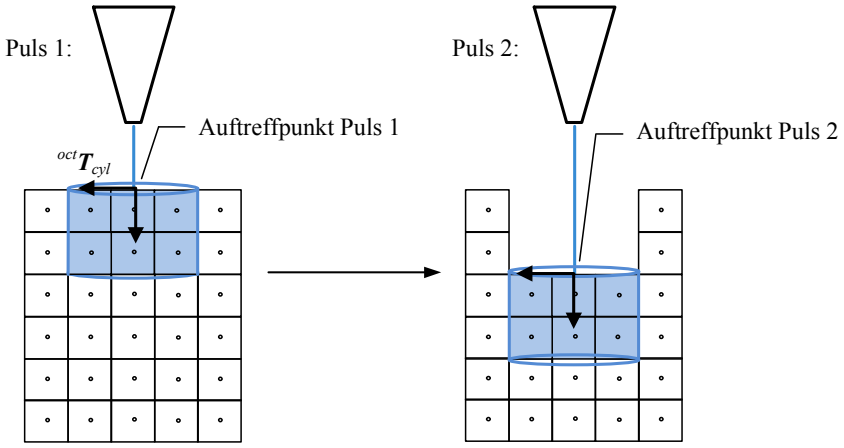


Abbildung 4.26: Bilden des Zylinders in Schritt 1 des Modells M_Z.

In Schritt 3 wird mit getIntersectingCubes() zuerst eine Voxelmenge $\mathbb{V}_{cube}$ vorausgewählt. Deren Voxel werden in Zylinderkoordinaten transformiert und es wird ihr Mittelpunkt gebildet (Gleichung 4.37 und 4.38), um auf Zugehörigkeit zum Zylinder geprüft zu werden (Gleichung 4.39 und 4.40). Für die daraus entstehende Voxelmenge der im Zylinder liegenden Voxel $\mathbb{V}_{cyl}$ wird in Schritt 4 die Senkung des Gesundheitszustands H berechnet. Schritt 4 des Modells M_Z unterscheidet sich dabei entscheidend vom Modell M_G. Bei M_G muss für jedes Voxel geprüft werden, ob das Voxel ein Oberflächen- oder ein verdecktes Voxel ist und abhängig davon der Gesundheitszustand berechnet werden. Im Modell M_Z wird nur unterschieden ob das Voxel vollständig innerhalb des Zylinders liegt oder am unteren Rand. Hierzu ist kein Raytracing nötig. Die in Abbildung 4.27 zu sehenden Voxel 1-6 sind vollständig im Zylinder, die Voxel 7-9 befinden sich am unteren Rand.

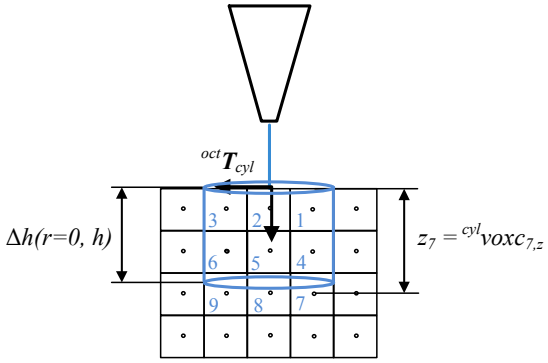


Abbildung 4.27: Zuordnung der Voxel zum Zylinder beim Modell M_Z.

Die Einteilung der Voxel vox_i aus $\mathbb{V}_{cyl}$ in innenliegende Voxel $vox_{i,Innen}$ oder Randvoxel $vox_{i,Rand}$ kann wie folgt über ihren Mittelpunkt ${}^{cyl}\boldsymbol{voxc}_i$ geschehen:

$$vox_i = \begin{cases} vox_{i,Innen} & \text{falls } (\Delta h_{max} - {}^{cyl}\, voxc_{i,z}) > voxsize/2 \\ vox_{i,Rand} & \text{falls } \left|(\Delta h_{max} - {}^{cyl}\, voxc_{i,z})\right| < voxsize/2 \end{cases} \tag{4.48}$$

Der Gesundheitszustand H der innenliegenden Voxel wird direkt auf 0 gesetzt, da diese auf jeden Fall vollständig eliminiert werden, der Gesundheitszustand der Randvoxel wird berechnet. Allgemein ist der Gesundheitszustand $H_{i,neu}$ nach der Berechnung wie folgt definiert:

$$H_{i,neu} = \begin{cases} H_{i,alt} - \left(\frac{\left|\Delta h(r{=}0,h) - {}^{cyl} voxc_{i,z}\right| + voxsize/2}{voxsize}\right) \cdot 100 & \text{wenn } vox_i = vox_{i,Rand} \\ 0 & \text{wenn } vox_i = vox_{i,Innen} \end{cases} \tag{4.49}$$

Zusammenfassung Die Gemeinsamkeiten und Unterschiede zwischen dem Modell Gaußscher Krater M_G und dem Zylindrischer Krater M_Z lassen sich im Folgenden zusammenfassen:

- Schritte 1-3 sind gleich, einziger Unterschied ist, dass der in Schritt 2 zu bildende Zylinder und damit die in Schritt 3 zu bildende Voxelmenge $\mathbb{V}_{cyl}$ beim Modell Gaußscher Krater wächst, beim Modell Zylindrischer Krater tritt dies nicht auf.

- In Schritt 4 ist beim Modell M_G für jedes Voxel aus $\mathbb{V}_{cyl}$ die Durchführung von Raytracing und eine aufwändige Berechung des neuen Gesundheitszustands nötig. Beim Modell M_Z ist kein Raytracing nötig, die Berechnung des neuen Gesundheitszustands ist weniger aufwändig.

M_G und M_Z stellen eine Implementierung des in Abschnitt 4.5.2 mathematisch beschriebenen Modells der Schnitttiefe eines Einzelpulsabtrags $\Delta h(r, h)$ (vgl. Gleichung 4.29) in einem Octree dar. Die Tiefenabhängigkeit dieser Schnitttiefe $\Delta h(r, h)$ wurde im Abschwächungskoeffizient $\mu(h)$ parametrisiert und implementiert. Die restlichen für Gleichung 4.29 nötigen Parameter α, w_0 und P_E wurden, wie in Abschnitt 4.5.2 genannt, aus der Literatur übernommen. Die Einzelpulsmodelle sollen in Kapitel 5 bezüglich Genauigkeit und Rechenzeit verglichen werden. Zum Lasern einer Linie muss kein eigenes Modell erstellt werden, da es durch Aneinanderreihen von Einzelpulsen entlang einer Trajektorie und damit durch die Einzelpulsmodelle M_G und M_Z beschrieben werden kann.

4.6 Präoperative Abtragssimulation und -visualisierung

Mit den in Abschnitt 4.5 implementierten Einzelpulsmodellen M_M kann eine Abtragssimulation durchgeführt und visualisiert werden. Dazu muss, wie in Abbildung 4.28 zu sehen ist, eine Trajektorie und ein Laserobjekt gewählt sein. Gegebenenfalls kann die Schnittnormale noch eingestellt werden. Außerdem müssen die Simulationsparameter Laserfrequenz f_p ($f_p \in \{10\ Hz; 20\ Hz\}$), Pulsenergie P_E ($P_E \in \{0,5\ J; 0,75\ J\}$) und das Einzelpulsmodell M_G oder M_Z gewählt werden. Die Berücksichtigung einer variablen Pulsenergie P_E im Modell ist vorgesehen, wurde aber nicht implementiert, so dass immer $P_E = 0,75\ J$ gilt. Der Strahldurchmesser w_0 kann nicht gewählt werden, er ist fest an den verwendeten Laser AT Fidelis angepasst ($w_0 = 0,55\ mm$ [4]). Die Einzelpulsmodelle können mit oder ohne Berücksichtigung der Debris und damit der Tiefenabhängigkeit der Schnitttiefe $\Delta h(r, h)$ (vgl. Gleichung 4.29) verwendet werden. Bei Verwendung mit Berücksichtigung der Debris nimmt der Abschwächungskoeffizient $\mu(h)$ mit steigender Tiefe h zu und es ergibt sich eine logarithmische Tiefenentwicklung $h(N)$ (vgl. Abbildung 4.20). Bei Verwendung ohne Debris bleibt μ ab dem ersten Puls konstant ($\mu = \mu(0) = konst.$) und die Tiefenentwicklung $h(N)$ verhält sich über die Anzahl von N Pulsen linear. Damit der Abtrag im Geoserver sichtbar ist, wird bei Auswahl des Laserobjekts dessen Geoserver-Oberflächenobjekt transparent dargestellt. Es wurden zwei Simulationsarten implementiert, die Simulationsart Punktpuls S_{PP} und die Simulationsart Linienpuls S_{LP}. Weder für S_{PP} noch für S_{LP} wurde eine quantitative Auswertung implementiert, es wird also z.B. nicht in einer GUI oder anderweitig grafisch angezeigt, wie viele Überfahrungen für das Durchlasern einer bestimmten Struktur nötig sind. Durch die

Visualisierung des Schnittprozesses kann dies aber qualitativ beurteilt werden. Zusätzlich wird die aktuelle Abtragstiefe im Terminal ausgegeben.

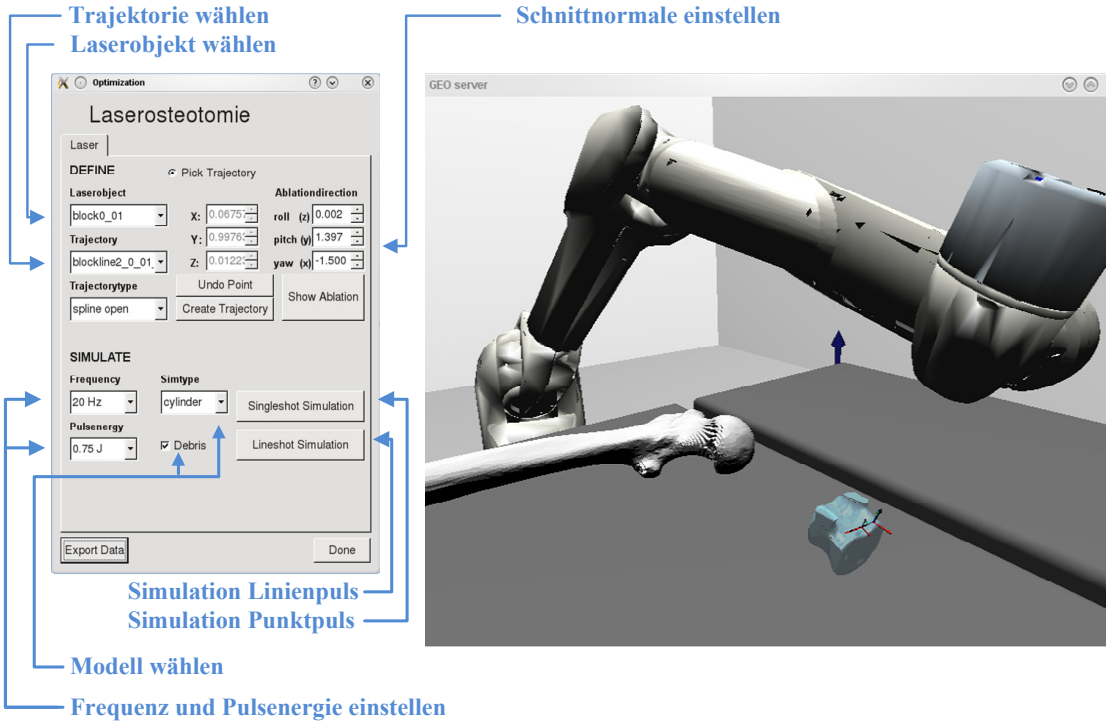


Abbildung 4.28: Einzelschritte zur Durchführung der präoperativen Simulation.

Simulationsart Punktpuls Bei der Simulationsart S_{PP} können einzelne Pulse auf einen Punkt des Laserobjekts eingebracht werden. Dadurch kann der Abtrag bezüglich der Frage beurteilt werden, ob mit den gewählten Laserparametern die gewünschte Abtragsgeschwindigkeit resultiert. Bei Betätigung des Knopfes in der Laserplanning GUI für die Simulationsart S_{PP} wird der TCP des Roboters ${}^{world}\boldsymbol{T}_{TCP}$ auf den mittleren Punkt der gewählten Trajektorie positioniert (vgl. Gleichung 4.50). Zur Umwandlung der kartesischen Trajektorienkoordinate in den Gelenkwinkelraum $\boldsymbol{Q}$ des Roboters (Gleichung 4.52) ist standardmäßig eine Inverskinematik ausgewählt, im vorangegangenen Arbeitsschritt Einrichten der OP Umgebung (vgl. Abschnitt 4.3) kann eine andere Inverskinematik ausgewählt und es können weitere Einstellungen dazu getätigt werden. Wie in Abbildung 4.29 sichtbar, liegen die Punkte der Trajektorie ${}^{world}\boldsymbol{T}_{Trajekt,i}$ in Worldkoordinaten $\boldsymbol{T}_{world}$ vor, die Inverskinematik arbeitet in Roboterbasiskoordinaten ${}^{world}\boldsymbol{T}_{RB}$. Daher muss der Trajektorienpunkt in Roboterbasiskoordinaten umgewandelt werden (Gleichung 4.51).

$$ {}^{world}\boldsymbol{T}_{TCP} = {}^{world}\boldsymbol{T}_{Trajekt,i} \tag{4.50} $$

$$ {}^{RB}\boldsymbol{T}_{TCP} = \left({}^{world}\boldsymbol{T}_{RB}\right)^{-1} \cdot {}^{world}\boldsymbol{T}_{TCP} \tag{4.51} $$

$$ {}^{RB}\boldsymbol{T}_{TCP} \xrightarrow{\text{invkin}} \boldsymbol{Q} \tag{4.52} $$

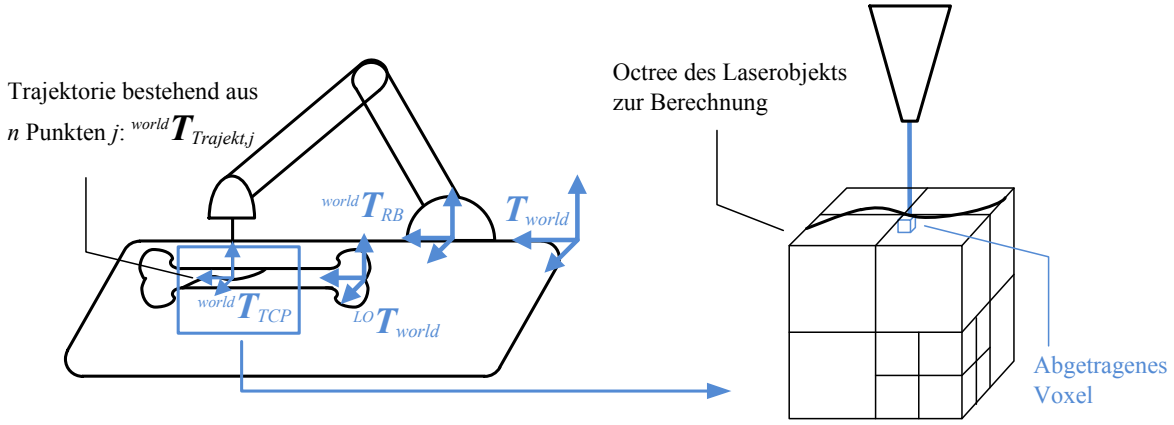


Abbildung 4.29: Funktionsprinzip der Simulationsart Punktpuls S_{PP}.

Danach wird ein einzelner Laserpuls in das Material eingebracht. Bei wiederholter Betätigung des Knopfes wird ein weiterer Puls eingebracht, so dass beobachtet werden kann, wie sich ein Krater an gewählter Stelle über mehrere Pulse entwickelt. Das Funktionsprinzip ist auf der rechten Seite von Abbildung 4.32 skizziert. Die Abbildung ist vereinfacht, bei Entfernung des hellblau dargestellten Voxels aus dem Octree des Gesamtvolumens zerfallen einige Octreeelemente in kleinere Teile, was hier nicht dargestellt wird. Je nach gewähltem Modell (M_G oder M_Z, mit oder ohne Debris) wird vom TCP aus der Abtrag im Octree des gewählten Laserobjekts berechnet. Dazu muss der in Weltkoordinaten vorliegende TCP ${}^{world}\boldsymbol{T}_{TCP}$ in Laserobjektkoordinaten ${}^{LO}\boldsymbol{T}_{TCP}$ umgerechnet werden:

$$ {}^{LO}\boldsymbol{T}_{TCP} = ({}^{world}\boldsymbol{T}_{LO})^{-1} \cdot {}^{world}\boldsymbol{T}_{TCP} \tag{4.53} $$

Der dann in Laserobjektkoordinaten vorliegende TCP ${}^{LO}\boldsymbol{T}_{TCP}$ wird verwendet, um gemäß Abschnitt 4.5.3 den Abtrag eines Pulses zu berechnen. Das im genannten Abschnitt verwendete Koordinatensystem des Octree ${}^{world}\boldsymbol{T}_{oct}$ entspricht dem hier genannten Koordinatensystem des Laserobjekts ${}^{world}\boldsymbol{T}_{LO}$.

Simulationsart Linienpuls Bei der Simulationsart S_{LP} können Pulse entlang der geplanten Trajektorie in das Laserobjekt eingebracht werden. Dabei kann der Abtrag bezüglich folgender Fragen beurteilen werden:

- Ist der Abtrag entlang der Trajektorie mit den gewählten Laserparametern durchführbar?
- Wie viele Überfahrungen sind notwendig, um eine bestimmte Abtragstiefe zu erzielen?
- Wie verhält sich die Tiefenentwicklung entlang der Trajektorie?
- Werden manche Stellen früher, manche Stellen nicht durchgelasert?
- Kann der verwendete Roboter mit der gewählten Inverskinematik den Laser entlang der Trajektorie führen?

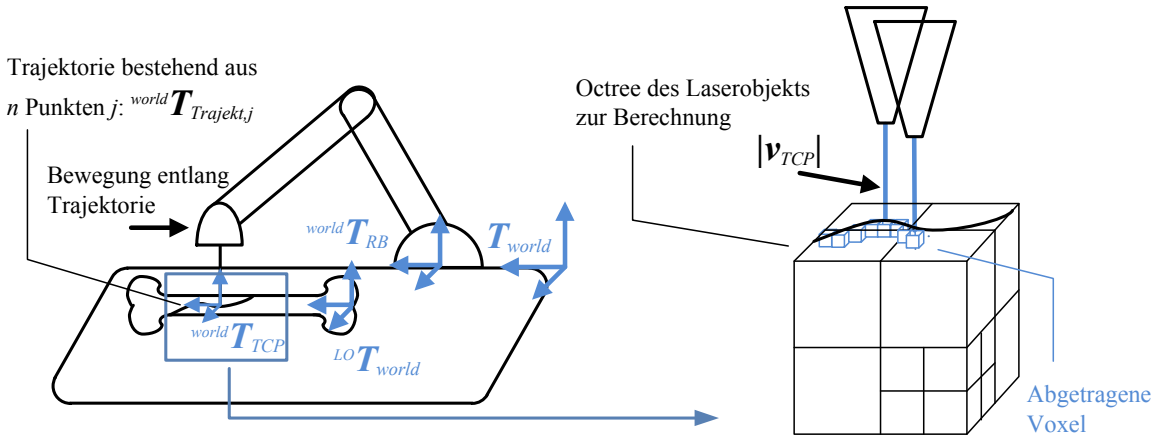


Abbildung 4.30: Funktionsprinzip der Simulationsart Linienpuls S_{LP}.

Bei Betätigung des Knopfes in der Laserplanning GUI für die Simulationsart S_{LP} wird der Roboter entlang der Trajektorie geführt. Da die Trajektorie, wie in Abschnitt 4.4 erklärt, durch Interpolation entstanden ist, besteht sie aus einzelnen Punkten P_j (bzw. ${}^{world}\boldsymbol{T}_{Trajekt,j}$), deren Verteilung von den Interpolatorparametern Taktzeit t_{Takt} und Maximale Geschwindigkeit $\dot{\Theta}_{i,max}$ jedes Freiheitsgrades Θ_i abhängen. Während bei der Simulationsart Punktpuls nur der mittlere dieser Punkte angefahren wird, werden hier alle Punkte nacheinander vom Roboter gemäß den Gleichungen 4.50 bis 4.52 angefahren. Wenn einer der Punkte wegen der Roboterkinematik nicht angefahren werden kann, wird dies im Terminal gemeldet und die Simulation wird abgebrochen. Auch das Setzen der einzelnen Pulse geschieht analog zur Simulationsart Punktpuls, nur dass nicht zwangsläufig an allen Punkten P_j der Trajektorie ein Einzelpuls gesetzt wird. An welchen Punkten der Linie ein Puls gesetzt wird, hängt von der Taktzeit t_{Takt} des Interpolators und der gewählten Frequenz f_p ab, was in Abbildung 4.31 verdeutlicht wird. In den dort dargestellten Fall gilt $t_{Takt} = \frac{1}{2 \cdot f_p}$, es wird also an jedem zweiten Punkt der Trajektorie ein Puls gesetzt und analog zur Simulationsart S_{PP} berechnet. Am Anfang und am Ende einer offenen splineförmigen Trajektorie gilt $|\boldsymbol{v}_{Start}| = |\boldsymbol{v}_{End}| = 0$, weshalb die Punkte P_j enger zusammen liegen. Dort werden auf engerem Raum auch mehr Pulse gesetzt.

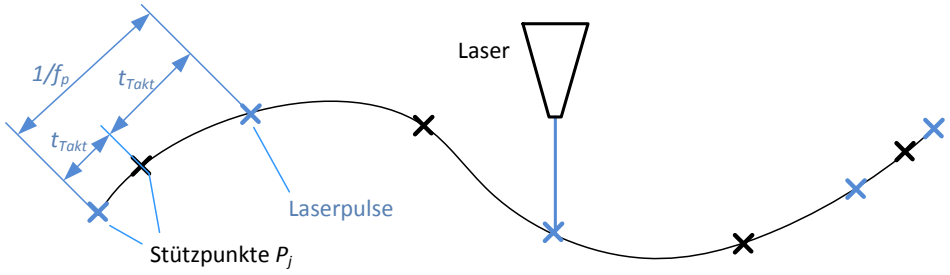


Abbildung 4.31: Pulsverteilung bei der Simulationsart Linienpuls S_{LP}.

Die hier genannte Taktzeit t_{Takt} des Interpolators entspricht nicht zwangsläufig der Zeit $t_{calc+vis}$, welche zur Berechnung t_{calc} und zur Visualisierung des Abtrags t_{vis} benötigt wird. Stattdessen hängt $t_{calc+vis}$ vom gewählten Modell M_M und der Rechenleistung des verwendeten Computers ab. Somit entspricht die Geschwindigkeit, mit der die Roboterbewegung während der Simulation im Geoserver dargestellt wird nur der realen Robotergeschwindigkeit, wenn gilt: $t_{Takt} = t_{calc+vis}$. Dies wird vom Bediener nicht beeinflusst.

Abtragsvisualisierung Wie in Abschnitt 4.4 beschrieben, beinhaltet die C++ Klasse Laserobjekt u.a. die drei folgenden Attribute: Ein Geoserver-Oberflächenobjekt zur Visualisierung des gesamten Objekts, welche der hier transparent dargestellte Oberfläche entspricht. Ein Volumenobjekt zum Speichern und zur Berechnung des gesamten Voxelvolumens, welches der Octreestruktur entspricht. Als drittes enthält die Klasse ein Geoserver-Voxelobjekt zur Visualisierung einzelner Voxel. Letzteres wird hier zur Darstellung des Abtrags, wie in Abbildung 4.32 zu sehen ist, verwendet. Statt bei einem entstehenden Krater bzw. einer Schnittfläche Voxel grafisch zu entfernen, wird dies hier mit dem Hinzufügen von Voxeln zur Visualisierung im Geoserver dargestellt. Diese Art der Darstellung wurde gewählt um nicht das Gesamte Voxelvolumen im Geoserver darstellen zu müssen. Bei mehreren 10.000 Voxeln wäre dies sehr rechenintensiv und unter Umständen nicht realisierbar. Eine andere Möglichkeit der Visualisierung ist, nur die geplante Schnittfläche als Voxelvolumen darzustellen und von dieser beim Wachsen des Kraters einzelne Voxel zu entfernen. Auch dabei müssten große Voxelmengen dargestellt werden, bei einer Abweichung des TCP von der vorgegebenen Trajektorie (vor allem relevant bei der später im Abschnitt 4.7 beschriebenen intraoperativen Abtragssimulation) könnten fälschlicherweise außerhalb des geplanten Volumens entfernte Voxel nicht visualisiert werden.

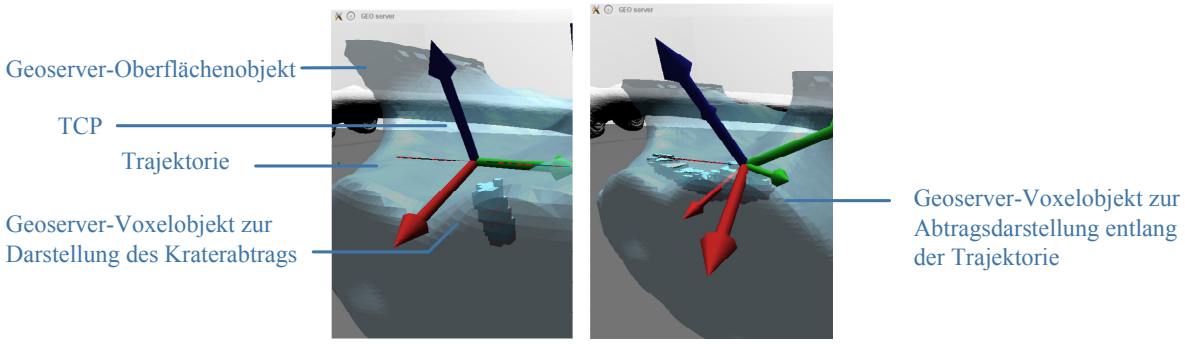


Abbildung 4.32: Visualisierung des Abtrags bei den Simulationsarten S_{PP} und S_{LP}.

4.7 Intraoperative Abtragsberechnung und -visualisierung

Im Anschluss an die Planung können die Daten an das reale System zur Ansteuerung von Roboter und Laser exportiert und die Laserosteotomie entsprechend durchgeführt werden. Während der Durchführung besteht eine Verbindung zwischen Operationsplanungssystem und realem System und es kann mit den realen Operationsparametern eine intraoperative Abtragsberechnung ähnlich der im vorherigen Abschnitt 4.6 erklärten präoperativen Simulation erfolgen. Die Verbindung zwischen Operationsplanungssystem und realem System soll in Zukunft bidirektional sein, d.h. es sollen sowohl die Aktordaten des realen Systems an das Operationsplanungssystem versendet werden, als auch Ergebnisse der Simulation vom Operationsplanungssystem an das reale System versendet werden. Im Rahmen dieser Arbeit wurde vorerst eine unidirektionale Verbindung implementiert, letztere Funktion ist somit nicht möglich. Zum Export der Daten wird nach vollständiger Planung der entsprechende Knopf (vgl. Abbildung 4.33) gedrückt.

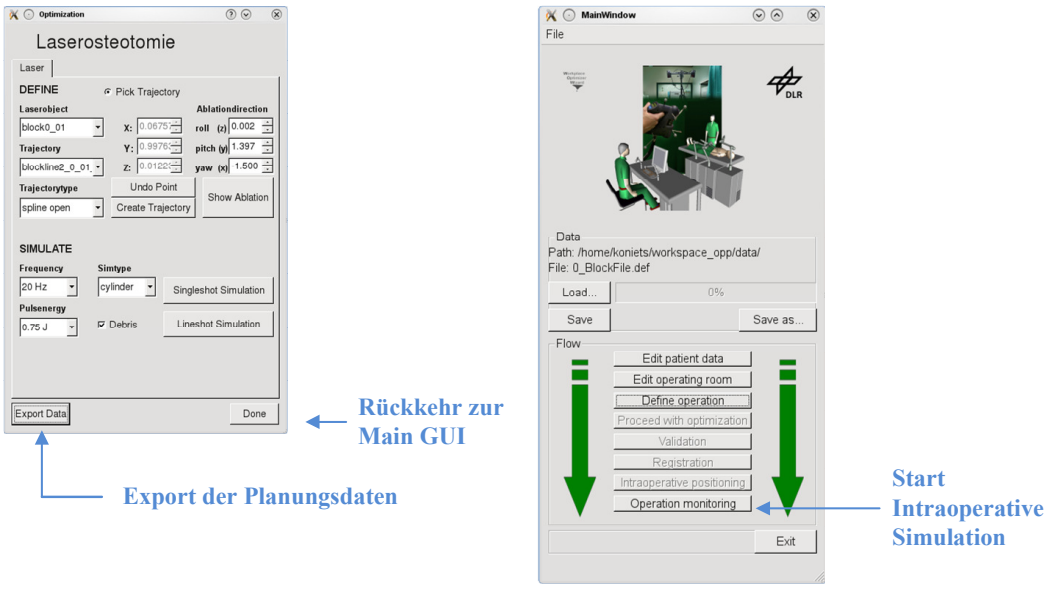


Abbildung 4.33: Einzelschritte in der GUI zum Export der Daten und zur intraoperativen Abtragsberechnung.

Beim Export der Daten wird eine Matlab m-Datei erzeugt, welches in das reale System zur Ansteuerung von Roboter und Laser eingelesen werden kann (eine beispielhafte m-Datei ist in Anhang B zu finden). Wie bereits in Abschnitt 4.1.2 erklärt, wird der Operationsroboter MIRO mit Simulink-Modellen angesteuert. Ein solches wurde für die Zwecke der Laserosteotomie angepasst und ist schematisch in Abbildung 4.34 zu sehen. Die aus der Planung exportierten Daten sind blau dargestellt. Die Ansteuerung des Lasers wurde nicht implementiert, der Laser wird daher vorerst im selben Simulink-Modell simuliert, mit welchem auch der MIRO angesteuert wird.

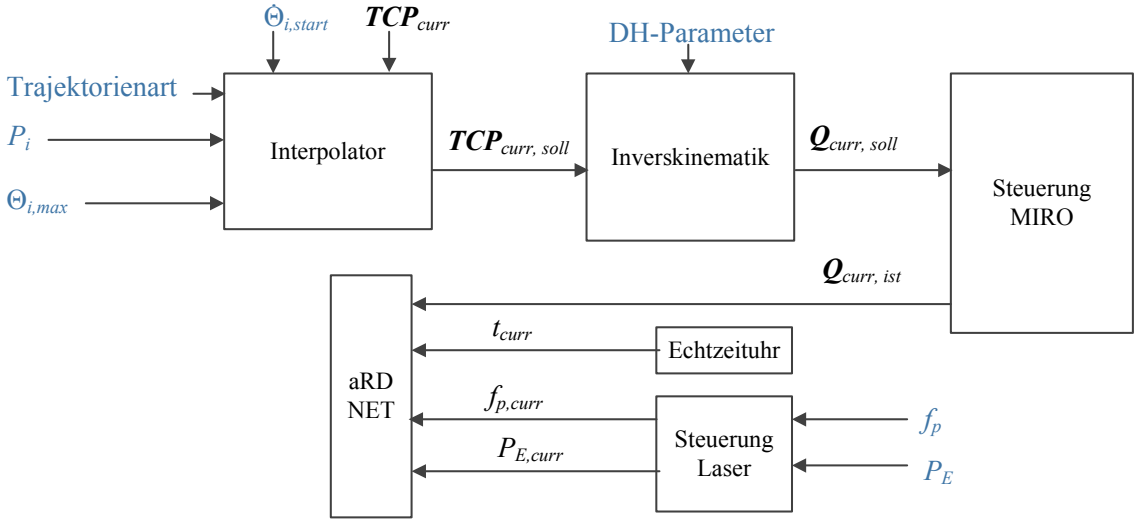


Abbildung 4.34: Schematische Darstellung des Simulink-Modells zur Ansteuerung des Roboters und zum Versenden der realen Prozessparameter zur intraoperativen Simulation im Operationsplanungssystem über aRD Netz.

Die exportierte m-Datei muss ausgeführt werden um die Parameter einzulesen. Diese beinhaltet folgende Parameter:

- DH-Parameter MIRO
- Pulsfrequenz f_p
- Pulsenergie P_E
- Punkte P_j der geplanten Trajektorie (bestehend aus Startpunkt, Endpunkt und den Stützpunkten) in Roboterbasiskoordinaten ${}^{RB}\boldsymbol{T}$
- Maximale Geschwindigkeit $\dot{\Theta}_{i,max}$ der translatorischen Freiheitsgrade Θ_i mit $i \in \{0; 1; 2\}$ auf der Trajektorie
- Abtragsnormale $\boldsymbol{n}$
- Gewählte Trajektorienform:
  - Trajektorienart 1: Lineare Verbindung zwischen den Punkten, offene Trajektorie
  - Trajektorienart 2: Lineare Verbindung zwischen den Punkten, geschlossene Trajektorie
  - Trajektorienart 3: Splineförmige Verbindung zwischen den Punkten, offene Trajektorie
  - Trajektorienart 4: Splineförmige Verbindung zwischen den Punkten, geschlossene Trajektorie
- Bei Trajektorienart 4 wird zusätzlich $\dot{\Theta}_i$ mit $i \in \{0; 1; 2\}$ im Startpunkt P_0 exportiert

Die DH-Parameter des Roboters wurden zuvor im Schritt Einrichten der OP Umgebung (vgl. Abschnitt 4.3) eingestellt und werden hier exportiert. Zur Änderungen der DH-Parameter kommt es, wenn z.B. ein anderer Laser mit unterschiedlichem Fokusabstand verwendet wird. Dann verschiebt sich der TCP des Roboters, und die DH-Parameter müssen angepasst werden, da sie in der Inverskinematik benötig werden. Die Pulsfrequenz f_p und die Pulsenergie P_E wurden während der präoperativen Abtragssimulation (vgl. Abschnitt 4.6) eingestellt und getestet. Da der Roboter gegebenenfalls anders am OP Tisch positioniert ist als in der Simulationsumgebung, müssen die Punkte der Trajektorie P_i in Roboterbasiskoordinaten angegeben werden. Unter diesen Umständen muss das Laserobjekt so zum Roboter positioniert sein, dass die Transformation in der Realität ${}^{RB}\boldsymbol{T}_{LO,real}$ und die Transformation in Planungsumgebung ${}^{RB}\boldsymbol{T}_{LO,plan}$ übereinstimmen oder es muss eine Registrierung geschehen. Ersteres ist in der Realität nicht zu erreichen. Bei einer Registrierung wird die Transformation ${}^{RB}\boldsymbol{T}_{LO,real}$ neu gebildet, indem z.B. am Roboter und am Laserobjekt optische Tracker angebracht werden und ${}^{RB}\boldsymbol{T}_{LO,real}$ durch eine Kamera vermessen wird [57]. Die Punkte der Trajektorie verschieben sich nach erfolgter Registrierung entsprechend der Differenz zwischen ${}^{RB}\boldsymbol{T}_{LO,real}$ und ${}^{RB}\boldsymbol{T}_{LO,plan}$. Da in diesem System keine Registrierung implementiert wurde, wird im Folgenden davon ausgegangen, dass ${}^{RB}\boldsymbol{T}_{LO,real} = {}^{RB}\boldsymbol{T}_{LO,plan}$ gilt.

Wie in Abschnitt 4.4 erklärt wurde, kommt zur Ansteuerung des Roboters derselbe Interpolator zum Einsatz, mit welchem die Trajektorie im Geoserver angezeigt wurde. Wenn sich bei der präoperativen Simulation keine Fehlermeldungen bezüglich der Roboterkinematik ergeben haben, ist die Trajektorie mit der gewählten Inverskinematik abfahrbar. Anders als in der präoperativen Simulation, bei der der Roboter direkt auf den Punkt P_{Start} der Trajektorie gesetzt wird, muss in der Realität zuerst von der momentanen Roboterposition TCP_{curr} auf P_{Start} der Trajektorie interpoliert werden. Aus diesem Grund ist TCP_{curr} ein weiterer Eingangsparameter in den Interpolator des Simulink-Modells (vgl. Abbildung 4.34). Bei den Trajektorienarten 1-3 kann diese Interpolation so erfolgen, dass für die Geschwindigkeit des Roboters im Punkt P_{start} gilt $|\boldsymbol{v}_{start}| = 0$. Bei der Trajektorienart 4 gilt am Punkt P_{start} $|\boldsymbol{v}_{start}| \neq 0$, weshalb bei dieser Interpolation von TCP_{curr} nach P_{start} auf die aus der Planung exportierten $\dot{\Theta}_i$ mit $i \in \{0; 1; 2\}$ beschleunigt werden muss. Dies ist in Abbildung 4.35 dargestellt.

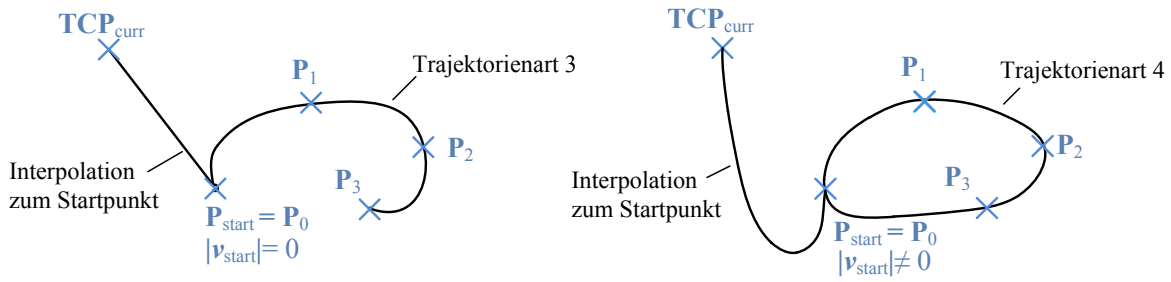


Abbildung 4.35: Interpolation von TCP_{curr} nach P_{start}. Links: Bei einer offenen splineförmigen Trajektorie. Rechts: Bei einer geschlossenen splineförmigen Trajektorie

Die Prozessparameter werden als Datenpaket per aRD Netz [8] an das Operationsplanungssystem versendet. Die Kommunikation ist unidirektional, asynchron und verwendet Shared Memories. Diese Datenpakete werden in vom Takt des Simulink-Modells bestimmten Zeitabständen in einem Shared Memory des Rechners, auf welchem die Operationsplanung läuft, abgelegt.

Die Datenpakete enthalten folgende Information:

- Aktuelle Zeit t_{curr}
- Aktueller TCP im Gelenkwinkelraum $\boldsymbol{Q}_{curr}$
- Aktuelle Pulsfrequenz $f_{p,curr}$
- Aktuelle Pulsenergie $P_{E,curr}$

Durch Betätigen des entsprechenden Schalters in der Main GUI (vgl. Abbildung 4.33) wird die intraoperative Simulation gestartet. Dabei wird zuerst eine aRD Netz Verbindung zu dem Shared Memory, in welchem die Datenpakete liegen, aufgebaut und die Daten werden zyklisch abgefragt. Wenn ein neues Datenpaket vorliegt, wird dieses an die Berechnung weitergegeben. Diese verläuft nach folgenden Einzelschritten:

- Ausrichten des Roboters mit $\boldsymbol{Q}_{curr}$
- Umrechnung des TCPs von $\boldsymbol{Q}_{curr}$ in ${}^{LO}\boldsymbol{T}_{TCP_{curr}}$
- Abtragsberechnung mithilfe ${}^{LO}\boldsymbol{T}_{TCP_{curr}}$, $f_{p,curr}$ und $P_{E,curr}$

Die Umrechnung von $\boldsymbol{Q}_{curr}$ in ${}^{LO}\boldsymbol{T}_{TCP_{curr}}$ geschieht mit einer Vorwärtskinematik und einer Transformation::

$$\boldsymbol{Q}_{curr} \xrightarrow{\text{fkin}} {}^{RB}\boldsymbol{T}_{TCP_{curr}} \tag{4.54}$$

$${}^{LO}\boldsymbol{T}_{TCP_{curr}} = {}^{LO}\boldsymbol{T}_{RB} \cdot {}^{RB}\boldsymbol{T}_{TCP_{curr}} \tag{4.55}$$

Die Übertragung der Daten vom Simulink-Modell zur OP Planung erfolgt in Echtzeit. Je nach gewähltem Einzelschussmodell M_G oder M_M und anderen Einflussfaktoren wie der Auflösung des Octrees, kann die Berechnung eines Einzelpulses jedoch länger dauern, als die Zeitdifferenz zwischen zwei aufeinanderfolgenden Datenpaketen. In diesem Fall werden in der OP Planung vom Simulink-Modell kommende Daten verpasst. Aus diesem Grunde wird bei jedem eintreffenden Datenpaket zuerst die Zeidifferenz zum letzten berechneten Datenpaket ermittelt.

$$\Delta t = t_{curr} - t_{last} \tag{4.56}$$

Mithilfe von Δt und $f_{p,curr}$ werden die Pulse mit konstantem Abstand zwischen dem letztem und aktuellem TCP in das Laserobjekt eingebracht. Je nach Dauer der Berechnung der Einzelpulse kommt es dabei zu der in Abbildung 4.36 sichtbaren Wegabweichung.

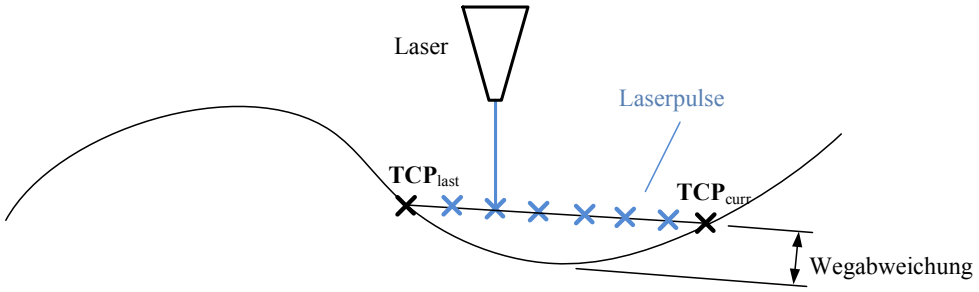


Abbildung 4.36: Interpolation zwischen zwei TCPs und die daraus resultierende Wegabweichung bei der intraoperativen Abtragsberechnung.

Nach der Berechung jedes Einzelpulses wird der Abtrag sowie die aktuelle Roboterpose im Geoserver dargestellt. Dies geschieht analog zur Visualisierung während der präoperativen Simulation (vgl. Abschnitt 4.6).

Kapitel 5

Experimente

Es wurden zwei Experimente durchgeführt. Das erste sollte die gemäß Abschnitt 4.5 implementierten Einzelpulsmodelle zylindrischer Krater M_Z und Gaußscher Krater M_G nach bestimmten Kriterien beurteilen, um eine Aussage über ihre Eignung für die Simulationsarten Punktpuls S_{PP} und Linienpuls S_{LP} zu treffen. Im zweiten Experiment wurde der Arbeitsablauf von der Planung bis zur intraoperativen Simulation durchgeführt und evaluiert. Bei den Experimenten wurden die in Abbildung 5.1 zu sehenden Laserobjekte verwendet, ihre Parameter sind in Tabelle 5.1 genannt.

Laserobjekte LO	Würfel W_1	Würfel W_2	Würfel W_3	Wirbel WB	Femur F_1	Femur F_2
Boundingbox $x \times y \times z\ [mm]$	$10\times10\times10$	$10\times10\times10$	$10\times10\times10$	$40\times43\times37$	$122\times75\times506$	$122\times75\times506$
Voxelgröße $voxsize\ [mm]$	$0,1$	$0,3$	$0,5$	$0,5$	$0,5$	$0,3$
Voxelanzahl $N\ [-]$	1157625	42875	9261	535050	37322972	169923825

Tabelle 5.1: Parameter der verwendeten Laserobjekte

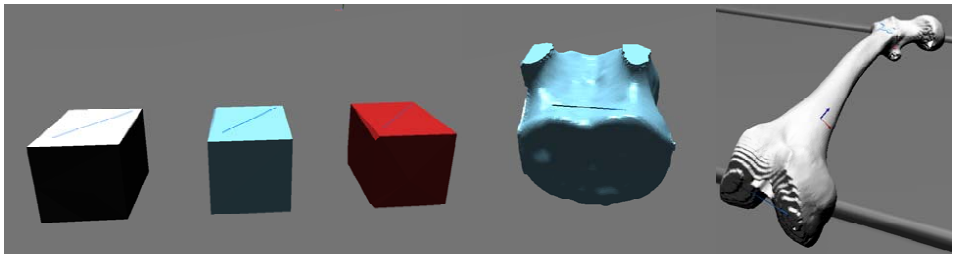


Abbildung 5.1: Verwendete Laserobjekte im Geoserver.

5.1 Experiment zur Beurteilung des Abtragsmodells

In diesem Experiment wurden zuerst, entsprechend der Simulationsart Punktpuls S_{PP}, mehrere Pulse mit den beiden Einzelpulsmodellen M_G und M_Z in ein Laserobjekt eingebracht und die Ergebnisse nach folgenden Kriterien beurteilt:

- Genauigkeit der Tiefenentwicklung
- Kraterform
- Rechenzeit

Die Genauigkeit und die Form wurden anhand der visualisierten Abtragsvoxel geprüft. Danach wurde entsprechend der Simulationsart Linienpuls S_{LP} Pulse entlang einer geplanten Trajektorie in ein Laserobjekt eingebracht und die Pulsverteilung, sowie die Form des entstehenden Abtragsvolumens betrachtet.

Genauigkeit Bereits zur Berechnung des verwendeten Abschwächungskoeffizienten $\mu(h)$ in Abschnitt 4.5 wurden die am DLR durchgeführten Experimente zum Knochenabtrag mit dem Er:YAG Laser AT Fidelis [4],[19] verwendet. In diesen wurde die Abtragstiefe bei N-fachem ($N \in \{2; 4; 16\}$) Überfahren einer Trajektorie auf einem Schweinefemur ermittelt. Unter den in Abschnitt 4.5 genannten Idealisierungen (konstante Pulsüberlappung resultierend aus konstanter Geschwindigkeit entlang der Trajektorie) kann die Abtragstiefe entlang der Trajektorie mit einer Kratertiefe $h_{exp}(N)$ bei Einbringen von N Pulsen auf dieselbe Stelle gleichgesetzt werden. Diese Experimente sollen auch hier verwendet werden um die Genauigkeit der Tiefenentwicklung der Modelle M_G und M_Z zu evaluieren. Dafür werden die Laserobjekte W_1, W_2 und W_3 verwendet. An jedem dieser Würfel W_i wurden Punktpulssimulationen mit den beiden Modellen M_M (Gaußscher Krater M_G und Zylindrischer Krater M_Z) in der Planungsumgebung durchgeführt. Es wurden $N \in \{2; 4; 8; 12; 16\}$ Pulse senkrecht auf eine Würfelfläche eingebracht, der Fokus lag dabei direkt auf der Oberfläche. Jede dieser Punktpulssimulationen wurde $j = 3$ mal durchgeführt und die entstandene Kratertiefe $h_{j,W_i,M_M}(N)$ gemittelt. Der Versuchsaufbau ist in Abbildung 5.2 zu sehen. Die abgetragenen Kratervoxel wurden exportiert und im Programm Amira manuell vermessen, wie auf der rechten Seite der Abbildung 5.2 zu sehen ist. Vollständig entfernte Voxel, deren Gesundheitszustand $H = 0$ beträgt, werden in hellgrauer Farbe dargestellt, nur teilweise entfernte Voxel, deren Gesundheitszustand $0 < H < 100$ entspricht dunkelgrau. Zum Messen der Kratertiefe $h_{j,W_i,M_M}(N)$ wurde der obere Punkt der Messlinie direkt auf die hellgraue Fläche gesetzt, der untere Messpunkt wurde zur Hälfte innerhalb der dunkelgrauen Grenzschicht positioniert, da diese eine nur zum Teil abgetragene Schicht darstellt.

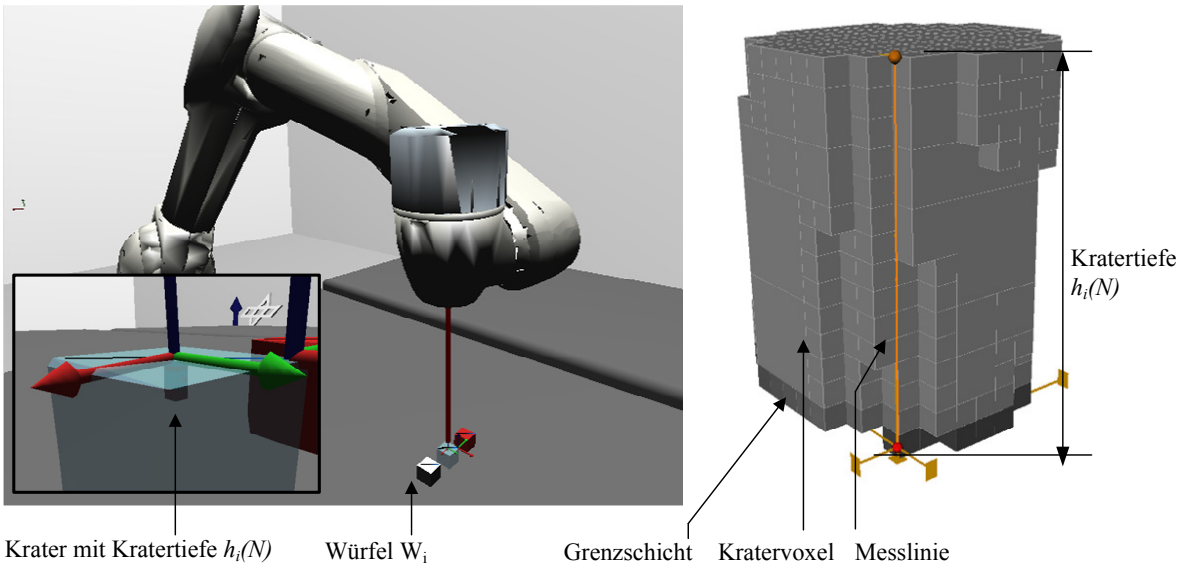


Abbildung 5.2: Links: Versuchsaufbau in der Planungsumgebung. Rechts: Vermessen des Kraters in Amira.

Die Ergebnisse zum Vergleich der Kratertiefe $\overline{h}_{W_i,M_M}(N)$ und $h_{exp}(N)$ sind in Tabelle 5.2 und in Abbildung 5.3 sowie Abbildung 5.4 zu sehen. Bei den Experimenten von [4],[19] wurden nur die Werte $h(2)$, $h(4)$, $h(16)$ ermittelt, die anderen Werte wurden hier über eine Matlab Fitting Curve angenähert. Die Werte $\overline{h}_{W_i,M_M}(N)$ entsprechen dem jeweiligen Mittelwert über $j = 3$ Versuche:

$$\overline{h}_{W_i,M_M}(N) = \frac{1}{3} \cdot \sum_{j=1}^{3} h_{W_i,M_M}(N) \tag{5.1}$$

Die Werte $\sigma_{W_i,M_M}(N)$ entsprechen der daraus resultierenden Standardabweichung:

$$\sigma_{W_i,M_M}(N) = \sqrt{\frac{1}{3-1} \cdot \sum_{j=1}^{3} \left(h_{j,W_i,M_M}(N) - \overline{h}_{W_i,M_M}(N) \right)^2} \tag{5.2}$$

In Tabelle 5.2 und Abbildung 5.3 sowie Abbildung 5.4 ist erkennbar, dass die Kratertiefe $\overline{h}_{W_i,M_Z}(N)$ beider Modelle M_G und M_Z bei hoher Auflösung, wie den hier verwendeten $0,1\ mm$ bei W_1, den Experimenten von [4],[19] folgen. Eine niedrige Auflösung wie die verwendeten $0,5\ mm$ resultieren vor allem bei M_Z in deutlichen Abweichungen. Bei einer Auflösung von $0,5\ mm$ liegt die Darstellung aller berechneten Werte $h \in]1,0\ mm; 1,5\ mm]$ immer bei $1,50\ mm$. Daher können Schnitttiefen einzelner Pulse von $\Delta h < voxsize$ nicht visualisiert werden. Bei niedriger Auflösung ist die Tiefenentwicklung des Modells M_G genauer als die des Modells M_Z.

$N\ [-]$	1	4	8	12	16
$h_{exp}(N)\ [mm]$	$0,28$	$0,85$	$1,27$	$1,46$	$1,58$
$\overline{h}_{W_1,M_Z}(N) + \sigma_{W_1,M_Z}(N)\ [mm]$	$0,35 + 0,10$	$0,93 + 0,18$	$1,35 + 0,20$	$1,52 + 0,25$	$1,57 + 0,20$
$\overline{h}_{W_1,M_G}(N) + \sigma_{W_1,M_G}(N)\ [mm]$	$0,28 + 0,06$	$0,88 + 0,06$	$1,32 + 0,06$	$1,55 + 0,00$	$1,62 + 0,06$
$\overline{h}_{W_2,M_Z}(N) + \sigma_{W_2,M_Z}(N)\ [mm]$	$0,45 + 0,00$	$1,05 + 0,00$	$1,55 + 0,09$	$1,75 + 0,17$	$1,95 + 0,00$
$\overline{h}_{W_2,M_G}(N) + \sigma_{W_2,M_G}(N)\ [mm]$	$0,15 + 0.00$	$0.90 + 0.15$	$1,25 + 0,17$	$1,45 + 0,17$	$1,65 + 0,00$
$\overline{h}_{W_3,M_Z}(N) + \sigma_{W_3,M_Z}(N)\ [mm]$	$0,25 + 0,00$	$1,25 + 0,00$	$1,75 + 0,00$	$2,25 + 0,00$	$2,25 + 0,00$
$\overline{h}_{W_3,M_G}(N) + \sigma_{W_3,M_G}(N)\ [mm]$	$0,25 + 0,00$	$0,75 + 0,00$	$1,25 + 0,00$	$1,25 + 0,00$	$1,75 + 0,00$

Tabelle 5.2: Ergebnisse des Vergleichs zwischen den von [4], [19] ermittelten Schnitttiefen $h_{exp}(N)$ und den Schnitttiefen $\overline{h}_{W_i,M_M}(N)$ der Modelle M_M angewandt auf die drei Würfel W_i angegeben mit der Standardabweichung $\sigma_{W_i,M_M}(N)$

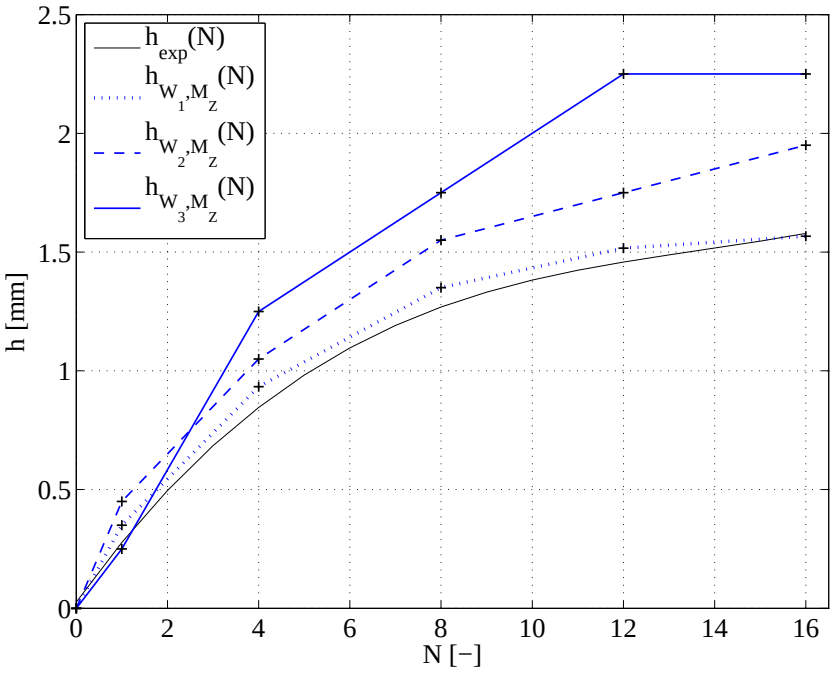


Abbildung 5.3: Kurven $h_{exp}(N)$ und $\overline{h}_{W_i,M_Z}(N)$ des Modells M_Z angewandt auf die drei Würfel W_i.

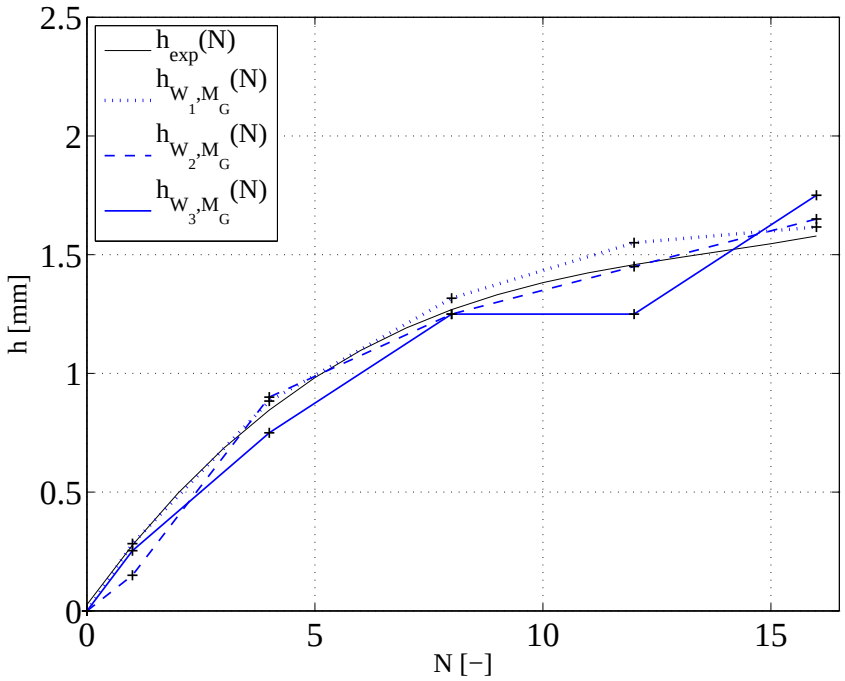
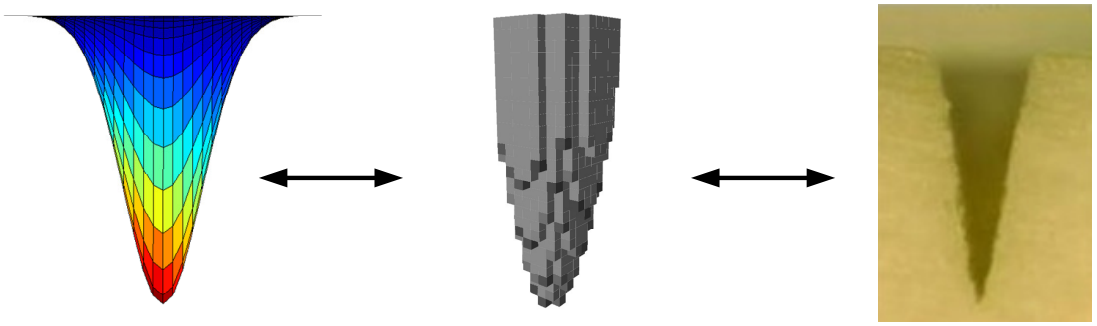


Abbildung 5.4: Kurven $h_{exp}(N)$ und $\overline{h}_{W_i,M_G}(N)$ des Modells M_G angewandt auf die drei Würfel W_i.

Kraterform Eine quantitative Beurteilung der Einzelpulsmodelle M_G und M_Z bezüglich ihrer Form ist nur schwer durchzuführen. Möglich wäre es den Voxelkrater mit einem mathematisch oder experimentell ermittelten Krater zu vergleichen und daraus ein quantitatives Maß für die Formübereinstimmung zu bilden, wie in Abbildung 5.5 dargestellt ist. Vor allem bei niedriger Auflösung von $0,5\ mm$ wäre dies nicht zielführend, da die charakteristische Kraterform in diesem Fall nicht mehr zu erkennen ist. Daher wurde hier vorerst qualitativ bewertet, wie gut ein durch die beiden Modelle M_Z und M_G berechneter Krater bei verschiedenen Auflösungen berechnet und visualisiert werden kann.



Möglichkeit 1:
Vergleich der Abtragsvoxel mit mathematisch berechnetem Krater

Möglichkeit 2:
Vergleich der Abtragsvoxel mit experimentell ermitteltem Krater

Abbildung 5.5: Möglichkeiten zur Beurteilung der Kraterform.

In Abbildung 5.6 sind Krater beider Modelle M_G und M_Z nach $N = 1$ und $N = 5$ in die Würfel W_i eingebrachten Pulsen zu sehen. Der Kraterradius betrug dabei $w_0 = 0,5\ mm$. Die charakteristische Kraterform ist nur bei der Voxelgröße $voxsize = 0,1\ mm$ zu erkennen. Sowohl die kreisförmige Querschnittsfläche wie die gaußsche Verjüngung des Kraters beim Modell M_G ist zu sehen. Das mittig spitze Zulaufen des Kraters von M_G ist bei $voxsize = 0,3\ mm$ nach $N = 5$ Pulsen noch zu erkennen, bei $voxsize = 0,5\ mm$ ist zwar eine Verjüngung erkennbar, die allerdings nicht in der Mitte liegt.

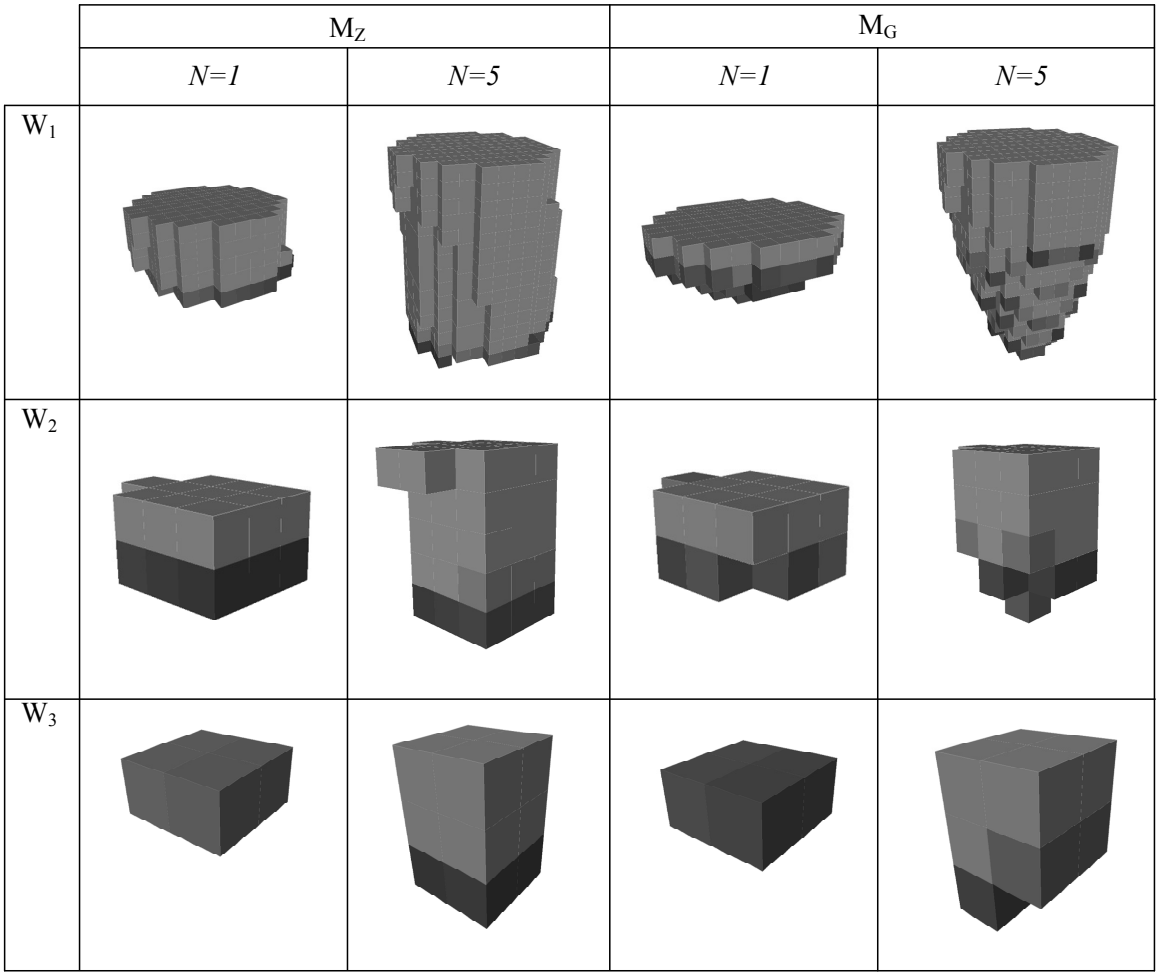


Abbildung 5.6: Kratervoxel der Modelle M_M angewandt auf die Würfel W_i.

Um zu prüfen, wie ein Krater berechnet und visualisiert wird, wenn der Laserstrahl $\boldsymbol{n}$ unter einem Winkel $\beta \neq 90°$ auf eine Fläche auftrifft, wurden Pulse mit einem Winkel von $\beta = 45°$ in die Würfel W_i eingebracht. Bei beiden Modellen treten dabei unterschiedliche Fehler auf, welche in den Abbildungen 5.7 und 5.8 zu sehen sind.

In Schritt 1 (vgl. Abschnitt 4.5.3) des Modells M_Z wird mit der Raytracingmethode getBeamIntersectingCubes() der Auftreffpunkt der optischen Achse gesucht. Während Schritt 2 wird von diesem Auftreffpunkt aus ein zur optischen Achse paralleler Zylinder mit der Tiefe Δh gebildet. Voxel, welche sich in dem Zylinder befinden, werden entfernt. Die in Abbildung 5.7 dargestellten überschüssigen Kratervoxel werden entfernt, da die Grundfläche des Zylinders

senkrecht zur optischen Achse des Lasers und damit senkrecht zur Zylinderachse orientiert ist, anstatt parallel zur bestrahlten Fläche des Laserobjekts. Damit entsteht beim ersten Puls ein fälschlicherweise um $45°$ orientierter Kraterboden, der sich bei darauffolgenden Pulsen nicht mehr ändert. Auch die dabei entstehende Abweichung $\Delta(\Delta h)$ von der Schnitttiefe Δh tritt nur beim ersten Puls auf und bleibt bei darauffolgenden Pulsen konstant. Dieser Fehler des Modells M_Z ist vom Winkel β zwischen Laserstrahl $\boldsymbol{n}$ und bestrahlter Fläche abhängig, nicht von der Orientierung des Octreekoordinatensystems ${}^{oct}\boldsymbol{T}_{world}$ zu $\boldsymbol{n}$.

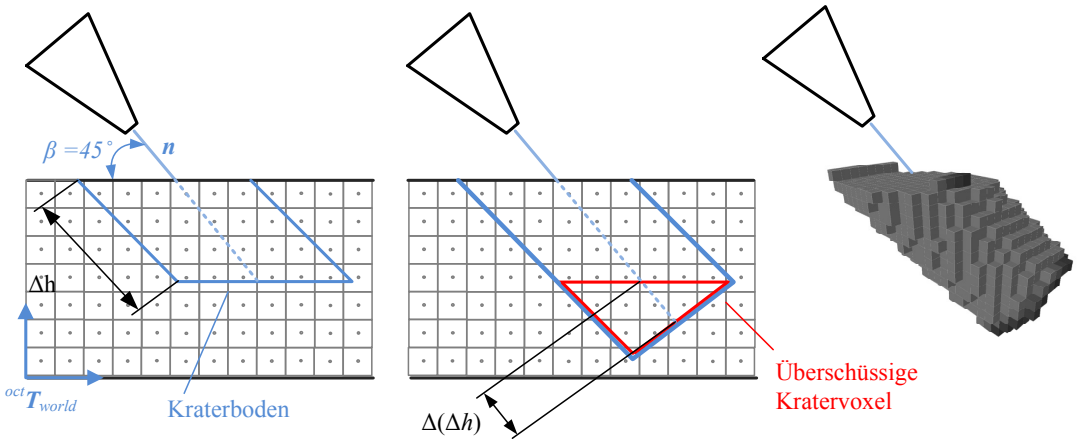


Abbildung 5.7: Fehler des Modells M_Z bei schrägem Auftreffen des Laserstrahls auf eine Fläche. Links: Gewünschte Kraterentwicklung. Mitte: Fehlerhafte Kraterentwicklung. Rechts: Fehlerhafte Kratervoxel

Der beim Modell M_G auftretende Fehler (vgl. Abbildung 5.8) ist vom Winkel β zwischen dem Octreekoordinatensystems ${}^{oct}\boldsymbol{T}_{world}$ und dem Laserstrahl $\boldsymbol{n}$ abhängig, nicht von der Orientierung der bestrahlten Fläche zu $\boldsymbol{n}$. Als β zwischen ${}^{oct}\boldsymbol{T}_{world}$ und $\boldsymbol{n}$ wird der Winkel bezeichnet, welcher zwischen $\boldsymbol{n}$ und der Oberfläche der bestrahlten Voxel liegt. Die Orientierung der Oberfläche der bestrahlten Voxel ist durch die Orientierung von ${}^{oct}\boldsymbol{T}_{world}$ bestimmt und entspricht nicht zwangsläufig der Orientierung der bestrahlten Gesamtoberfläche. In der Abbildung 5.8 sind die bestrahlte Voxeloberfläche und die bestrahlte Gesamtoberfläche jedoch gleich orientiert. Wie in Abschnitt 4.5.3 erklärt, werden in Schritt 3 des Modells M_G die Voxel gesucht, welche in einem zum Laserstrahl $\boldsymbol{n}$ parallelen Zylinder liegen. Die Zylindertiefe setzt sich zusammen aus der vom Auftreffpunkt in Strahlrichtung resultierenden maximalen Schnitttiefe Δh_{max} und dem Abstand des Auftreffpunkts zum Strahlursprung. Der in der Mitte der Abbildung 5.8 zu sehende rot ausgefüllte Bereich wird dabei nicht berücksichtigt. Der Rest des rot umrandeten Bereichs geht in Schritt 4 des Modells verloren. In diesem Schritt werden im Zylinder liegende Voxel in verdeckte und nicht verdeckte Voxel unterteilt. Davon abhängig wird ihr Gesundheitszustand H neu berechnet, wobei H der Oberflächenvoxel durch Gleichung 4.29 berechnet wird und H der verdeckten Voxel vom Energiezustand des darübeliegenden Voxels abgeleitet wird. Das darüberliegende Voxel wird bestimmt, indem vom betroffenen Voxel die Raytracingmethode getBeamIntersectingCubes() in entgegengesetzter Strahlrichtung angewendet und das erste getroffene Voxel als darüberliegendes Voxel definiert wird. Dies ist in Abbildung 5.9 dargestellt. Bei $\beta = 90°$ ist das darüberliegende Voxel klar definiert, bei $\beta = 45°$ liegen die Voxel 1, 2 und 3 über dem zu berechnenden Voxel. Welches Voxel von getBeamIntersectingCubes() als erstes gefunden wird und in Schritt 4 als darüberliegendes Voxel definiert wird, ist damit von β zwischen

${}^{oct}\boldsymbol{T}_{world}$ und $\boldsymbol{n}$ abhängig. Bei $\beta \in]45°; 90°]$ wird Voxel 1 getroffen, bei $\beta \in [0°; 45°]$ Voxel 3. Wenn immer Voxel 1 getroffen wird, kommt es am oberen Randrand zu der in Abbildung 5.8 sichtbaren senkrecht abfallenden Kraterwand, wenn immer Voxel 3 getroffen wird, entsteht am unteren Kraterrand eine waagrechte Kraterwand. Um eine Unabhängigkeit der Methode M_G zu erreichen, müsste in Schritt 4 die Restenergie von verdeckten Voxeln nicht von nur einem darüberliegenden, sondern in gewissen Fällen von mehreren darüberliegenden Voxeln berechnet werden.

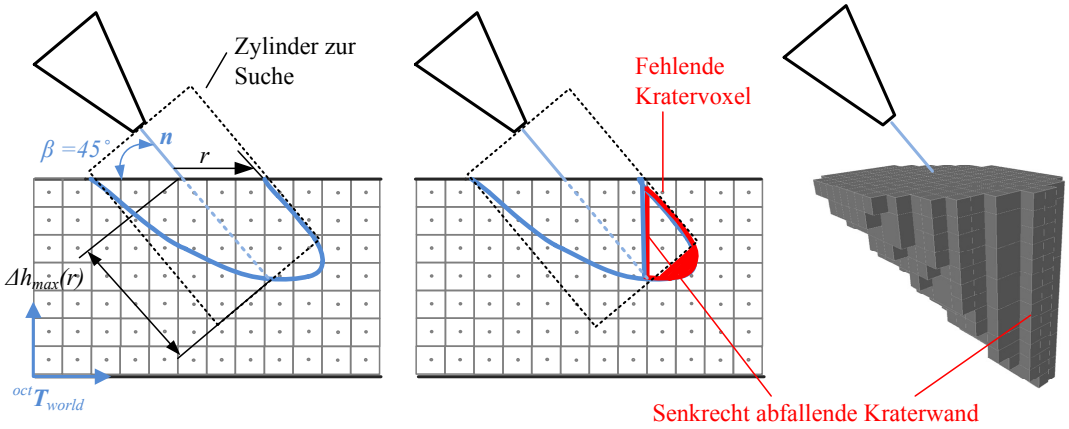


Abbildung 5.8: Fehler des Modells M_G bei Laserstrahlen $\boldsymbol{n}$, welche nicht senkrecht zur Voxeloberfläche orientiert sind. Links: Gewünschte Kraterentwicklung. Mitte: Fehlerhafte Kraterentwicklung. Rechts: Fehlerhafte Kratervoxel.

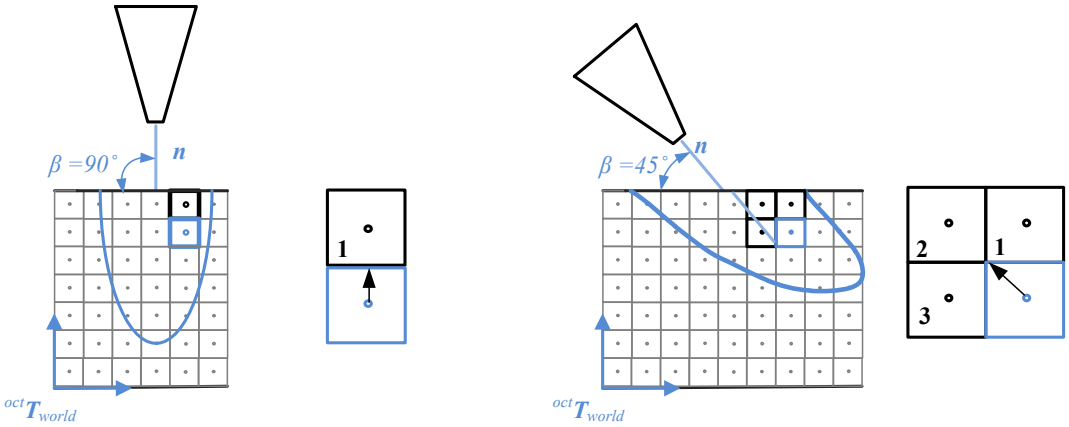


Abbildung 5.9: Raytracing bei Schritt 4 des Modells M_G mit $\beta = 90°$ und $\beta \neq 90°$.

Rechenzeit Als nächstes wurden die Modelle M_M während der Simuationsart S_{PP} bezüglich der Simulationszeit t_{sim} geprüft. Diese setzt sich zusammen aus t_{calc}, der Zeit zur Berechnung der Abtragsvoxel durch das jeweilige Modell M_M, und t_{vis}, der Zeit zum Visualisieren der Voxel im Geoserver.

$$t_{sim} = t_{calc} + t_{vis} \tag{5.3}$$

t_{calc} hängt ab vom Modell M_M, von der Voxelgröße $voxsize$, von der Anzahl der Voxel eines Laserobjekts N und von der Kratertiefe h und wurde nach diesen Kriterien getestet. t_{vis} hängt von der Anzahl N der Voxel ab, welche nach Berechnung eines Pulses zur Visualisierung an den Geoserver gesendet werden. Bei einheitlichen Laserparametern wird der nach einem Puls entstehende Krater bei verschiedenen Voxelgrößen $voxsize$ mit einer unterschiedlichen Anzahl von Voxeln N dargestellt. Daher wurde t_{vis} in Abhängigkeit von N geprüft. Diese Laufzeittests wurden auf einem Computer mit dem Betriebssystem Linux Suse und vier Intel(R) Core(TM)2 Quad CPU Q9550, 2.83GHz Prozessoren durchgeführt. Zum Erzeugen des C++-Programms wurde der GNU C++ Kompilierer verwendet, die Optimierungsoption des Kompilierers wurde nicht verwendet. Da Linux Suse im Mehrprozessbetrieb arbeitet, bestand während den Tests keine Kontrolle darüber, welche Prozesse neben dem zu testenden noch liefen. Daher wurde zuerst jede Messung von t_{calc} und t_{vis} 500 mal durchgeführt und die vorübergehenden Mittelwerte $\tilde{\bar{t}}_{calc}$ und $\tilde{\bar{t}}_{vis}$ gebildet. Danach wurden aus den 500 Messwerten diejenigen als ungültig gewertet, welche $\tilde{\bar{t}}_{calc}$ bzw. $\tilde{\bar{t}}_{vis}$ um ein doppeltes übertrafen. Von den übrig bleibenden Werten wurden der Mittelwert $\bar{t}_{calc}$ bzw. $\bar{t}_{vis}$ und die Standardabweichung σ_{calc} bzw. σ_{vis} gebildet. In Tabelle 5.3 sind die Ergebnisse der Rechenzeit t_{calc,M_M} beider Modelle in Abhängigkeit von der Voxelgröße $voxsize$ dargestellt. Dabei wurden einzelne Pulse in die Laserobjekte W_1, W_2 und W_3 bei einer Kratertiefe von $h = 0$ eingebracht. Unter diesen Umständen ist die Rechenzeit bei M_G höher als bei M_Z, jedoch nur geringfügig. Mit den verwendeten Laserprozessparametern ($w_0 = 0,55\ mm$, $\Delta h \approx 0,3\ mm$) ergeben sich bei einer Voxelgröße von $voxsize = 0,1\ mm$ ca. 360 Kratervoxel, bei $voxsize = 0,3\ mm$ ca. 36 Kratervoxel und bei $voxsize = 0,5\ mm$ nur noch ca. 4 Kratervoxel. Damit ist der Anstieg in t_{calc} bei steigender Auflösung zu begründen. Eventuell gibt es beim Übergang von $voxsize = 0,3\ mm$ auf $voxsize = 0,1\ mm$ ein Speicherproblem, anders ist der ca. 500-fache Anstieg in t_{calc} nicht zu erklären.

$voxsize\ [mm]$	$0,1$	$0,3$	$0,5$
$\bar{t}_{calc,M_G}(voxsize) + \sigma_{M_G}(voxsize)\ [ms]$	$251,392 \quad + \\ 0,410$	$0,357 \quad + \\ 0,013$	$0,107 \quad + \\ 0,005$
$\bar{t}_{calc,M_Z}(voxsize) + \sigma_{M_Z}(voxsize)\ [ms]$	$197,727 \quad + \\ 0,569$	$0,322 \quad + \\ 0,009$	$0,072 \quad + \\ 0,004$

Tabelle 5.3: Rechenzeit t_{calc,M_M} in Abhängigkeit von der Voxelgröße $voxsize$ der Laserobjekte

Die Ergebnisse der Rechenzeit t_{calc,M_M} beider Modelle in Abhängigkeit von der Gesamtanzahl N der Voxel des jeweiligen Laserobjekts sind in Tabelle 5.4 zu sehen. Es wurden einzelne Pulse in die Laserobjekte W_3, WB und F_1 bei einer Kratertiefe von $h = 0$ eingebracht. Alle Laserobjekte haben dieselbe Voxelgröße $voxsize = 0,5\ mm$. Während die Unterschiede zwischen M_G und M_Z bei den Laserobjekten W_1 und F_1 zu vernachlässigen sind, tritt beim Laserobjekt

WB ein ca. 35-facher Unterschied in t_{calc} auf, welcher nicht erklärt werden konnte. Entgegengesetzt der Annahme, dass mit steigendem N auch t_{calc} steigt, liegt die Rechenzeit des Modells M_G beim Laserobjekt F_1 unter der von WB. Auch dieser Effekt konnte nicht geklärt werden. Eventuelle Gründe könnten sein, dass das F_1 bezüglich seiner Geometrie weniger komplex ist als WB und daher die octreeinternen Suchalgorithmen (getBeamIntersectingCubes() und getIntersectingObjects()) schneller zu einem Ergebnis kommen. Um dies zu Prüfen müsste der Test mit Laserobjekten gleicher Komplexität ähnlich der drei Würfel W_i wiederholt werden.

$N\ [-]$	9261	535050	37322972
$\bar{t}_{calc,M_G}(N) + \sigma_{M_G}(N)\ [ms]$	$0,107 + 0,005$	$69,842 + 0,257$	$45,599 + 0,892$
$\bar{t}_{calc,M_Z}(N) + \sigma_{M_Z}(N)\ [ms]$	$0,072 + 0,004$	$2,071 + 0,040$	$45,433 + 1,338$

Tabelle 5.4: Rechenzeit t_{calc,M_M} in Abhängigkeit von der Gesamtvoxelanzahl N der Laserobjekte

In Tabelle 5.5 sind die Ergebnisse der Rechenzeit t_{calc,M_M} beider Modelle in Abhängigkeit von der Kratertiefe h dargestellt. Dabei wurden einzelne Pulse in die Laserobjekte W_1, W_2 und W_3 bei den unterschiedlichen Kratertiefen $h = 0,0\ mm$, $h = 0,5\ mm$, $h = 1,0\ mm$ und $h = 1,5\ mm$ eingebracht. Wie in Abschnitt 4.5.3 erklärt, wächst beim Modell M_M der im Octree zu durchsuchende Bereich mit wachsender Kratertiefe h. Dies ist an den Ergebnissen dieses Tests deutlich erkennbar. Während t_{calc,M_Z} bei steigendem h nahezu konstant bleibt, wächst t_{calc,M_G} von $0,3\ ms$ bei $h = 0,0\ mm$ auf $32\ ms$ bei $h = 1,5\ mm$.

$h\ [mm]$	$0,0$	$0,5$	$1,0$	$1,5$
$\bar{t}_{calc,M_G}(h) + \sigma_{M_G}(h)\ [ms]$	$0,357 + 0,013$	$1,261 + 0,007$	$3,535 + 0,044$	$32,302 + 0,117$
$\bar{t}_{calc,M_Z}(h) + \sigma_{M_Z}(h)\ [ms]$	$0,322 + 0,009$	$0,496 + 0,010$	$0,314 + 0,008$	$0,525 + 0,008$

Tabelle 5.5: Rechenzeit t_{calc,M_M} in Abhängigkeit von der Kratertiefe h

Die Ergebnisse der Messungen von t_{vis}, der Zeit zum Visualisieren der Voxel im Geoserver, sind in Abbildung 5.6 zu sehen. Diese Messung hängt nicht vom Modell M_M ab, sondern nur von der Anzahl N der zu visualisierenden Voxel. Mit dem Modell M_Z wurden daher Pulse ($w_0 = 0,5\ mm$, $h \approx 0,3\ mm$) in die Würfel W_i eingebracht, wodurch sich je nach Auflösung der Würfel Voxelmengen von 4, 36 und 360 ergaben. Der Anstieg von t_{vis} über N verhält sich dabei nahezu linear.

$N\ [-]$	4	36	360
$\bar{t}_{vis,M_Z}(N) + \sigma_{M_Z}(N)\ [ms]$	$0,068 + 0,006$	$0,381 + 0,111$	$29,522 + 19,492$

Tabelle 5.6: Zeit zur Visualisierung t_{vis,M_Z} des Kraters in Abhängigkeit von der Anzahl der zu visualisierenden Kratervoxel N

Simulationsart Linienpuls Bei der Simulationsart Linienpuls S_{LP} werden Einzelpulse entlang einer Trajektorie in das Laserobjekt eingebracht. Mit diesem Experiment wurde die Eignung der beiden Modelle M_G und M_Z zur Simulationsart S_{LP} getestet. Es wurde die Form des visualisierten Voxelvolumens und die Verteilung der Pulse entlang der Trajektorie beurteilt. Dazu wurden die Würfel W_1 ($voxsize_1 = 0,1\ mm$), W_2 ($voxsize_2 = 0,3\ mm$) und W_3 ($voxsize_3 = 0,5\ mm$) verwendet. Auf dem Würfel W_1 wurden zwei Punkte $P0$ und $P1$ eingezeichnet und zwischen diesen beiden Punkten mithilfe des kartesischen Interpolators eine Trajektorie der Trajektorienart 1 (vgl. Abschnitt 4.4) erzeugt. Das Einzeichnen der Punkte geschah manuell im Planungssystem gemäß des in Abschnitt 4.4 erklärten Arbeitsschrittes. Mit der Taktzeit $t_{Takt} = 0,05\ s$, der Maximalgeschwindigkeit der drei translatorischen Freiheitsgrade $\dot{\Theta}_{i,max} = 0,2$ für $i \in \{0;1;2\}$ und einer Trajektorienlänge von $11,27\ mm$ ergaben sich 48 Interpolationspunkte P_j. Da der verwendete Interpolator von einem Gelenkwinkelinterpolator abgeleitet ist, handelt es sich bei $\dot{\Theta}_{i,max} =$ nicht um $0,2\ \frac{m}{s}$ sondern um $0,2\ \frac{rad}{s}$. Zur Verwendung zur kartesischen Interpolation musste der Wert durch Ausprobieren eingestellt werden. Die eingestellte Frequenz betrug $f_p = 20\ Hz$, was bedeutet, dass an jedem Punkt P_j ein Puls in das Laserobjekt eingebracht wurde. Der Versuchsaufbau ist in Abbildung 5.10 zu sehen. Es wurde auf allen Würfeln W_i entlang derselben Trajektorie eine Überfahrung gelasert. Da die Trajektorie auf W_1 eingezeichnet wurde, musste sie zur Verwendung für W_2 und W_3 entsprechend verschoben werden.

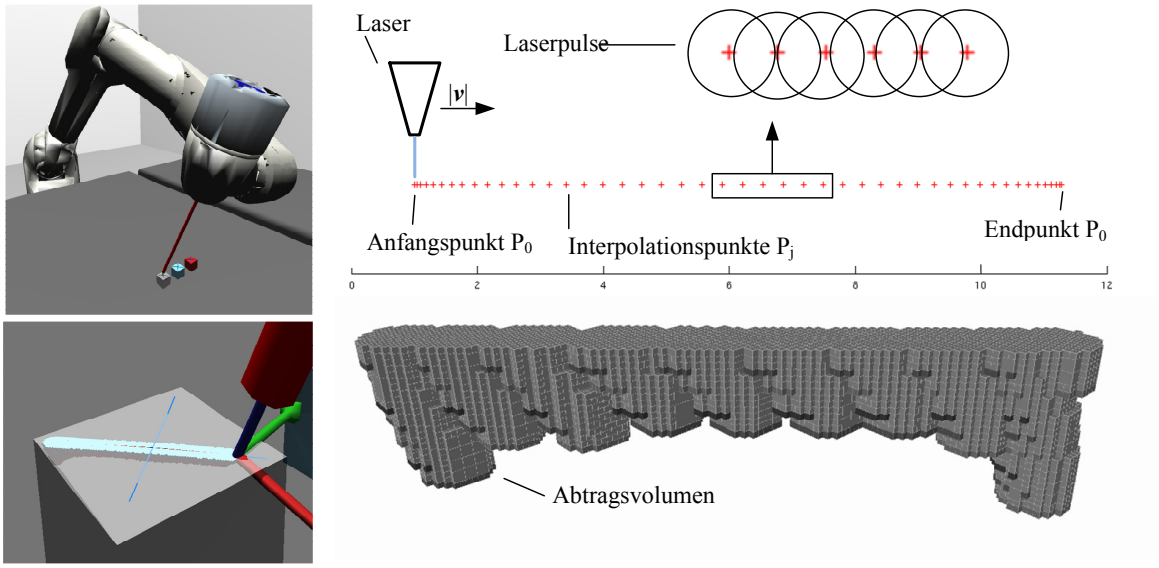


Abbildung 5.10: Aufbau des Experiments zur Simulationsart Linienpuls. Links: Ansicht im Geoserver. Rechts: Darstellung der Pulsverteilung auf der Trajektorie.

In Abbildung 5.11 sind die Ergebnisse zu sehen. Die Abtragsvolumina wurden rein qualitativ beurteilt. Auch hier wird deutlich, dass zur Erkennung einer charakteristischen Form des Abtragsvolumens eine Auflösung von $voxsize = 0,1\ mm$ nötig ist. Bei dieser Auflösung können sowohl bei M_G als auch bei M_Z die Einzelpulse entlang der Trajektorie erkannt werden. Bei M_G ist darüber hinaus die charakteristische Verjüngung der Schnittkante bei steigender Tiefe h sichtbar. Diese Verjüngung ist auch bei $voxsize = 0,03\ mm$ noch sichtbar, jedoch stark vereinfacht. Es ist bei allen Voxelgrößen erkennbar, dass sich durch die unterschiedliche Pulsdichte (Pulse pro Trajektorienabschnitt) entlang der Trajektorie eine wegabhängige Tiefenentwicklung ergibt. Am Anfang und am Ende fallen mehr Pulse, das Voxelabtragsvolumen ist dort entsprechend tiefer.

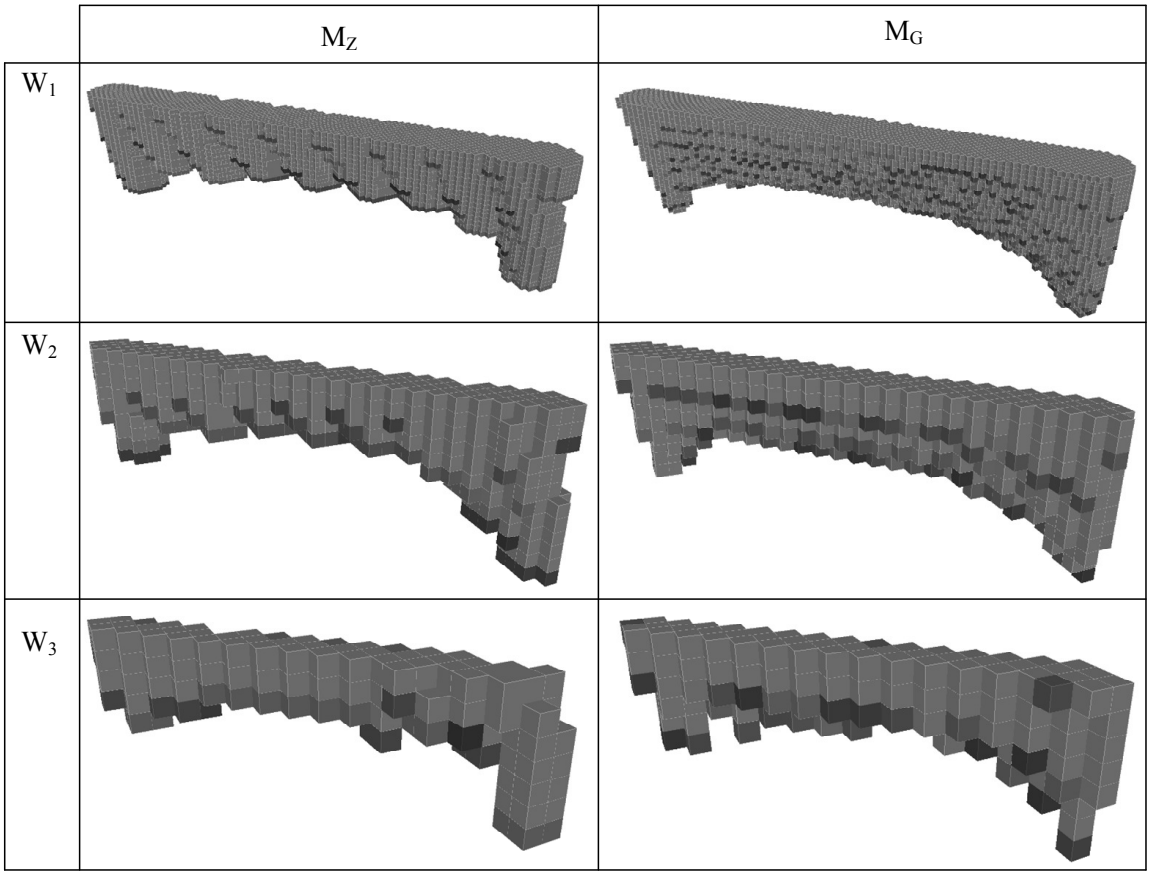


Abbildung 5.11: Voxelabtragsvolumen der Modelle M_M entlang der Trajektorie angewandt auf die Würfel W_i.

Bei Verwendung von M_Z zur Simulationart S_{LP} tritt ein Fehler auf, welcher in Abbildung 5.12 zu sehen ist. Auf der linken Seite der Abbildung ist die gewünschte Kraterentwicklung zu sehen. Der Teil von Puls 2, welcher sich mit Puls 1 überschneidet, resultiert in einer Absenkung des Kraterbodens von Puls 1, der Teil, welcher sich nicht überschneidet senkt die vorher von Puls 1 nicht bestrahlte Oberfläche des Laserobjekts ab. Der Fehler ist in der Mitte und auf der rechten Seite der Abbildung zu sehen. Der Zylinder von Puls 2 wird ab dem Punkt gebildet, in welchem die optische Achse mit der Raytracingfunktion getBeamIntersectingCubes() auf das Laserobjekt auftrifft. Dabei kommt es zu einer Hinterschneidung im Laserobjekt.

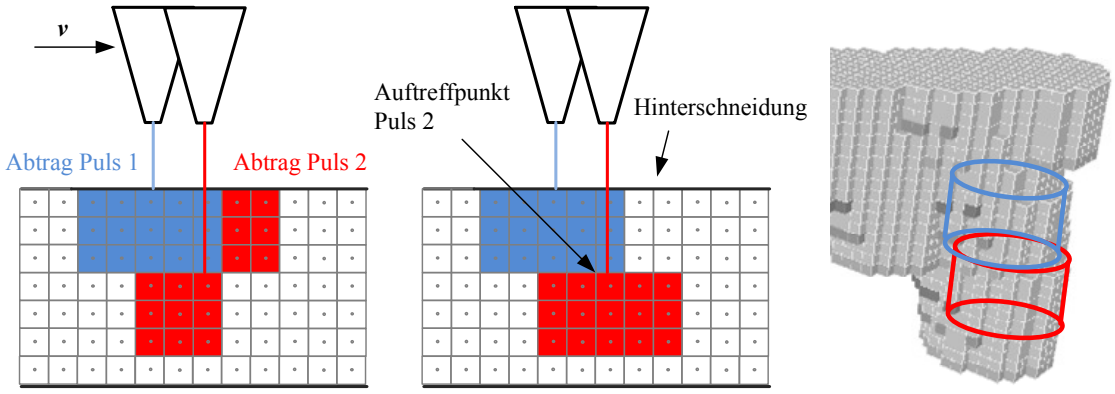


Abbildung 5.12: Fehler des Modells M_Z bei der Simulationsart S_{LP}. Links: Gewünschte Kraterentwicklung. Mitte: Fehlerhafte Kraterentwicklung, Rechts: Fehlerhafte Kratervoxel.

Zusammenfassung des Experimentes Die Experimente zur Genauigkeit und zur Form zeigen, dass zur Visualisierung einzelner Pulse im Bereich der für den Hartgewebeabtrag üblichen Kraterdimensionen ($\Delta h \approx 0,3\ mm$, $w_0 \approx 0,55\ mm$) Voxelgrößen von $voxsize < 0,3\ mm$ nötig sind. Die Genauigkeit der Visualisierung ist bei niedrigeren Auflösungen, wie den verwendeten $0,3\ mm$ und $0,5\ mm$, ungenügend, darüber hinaus sind die zylindrische oder gaußsche Kraterform nicht mehr zu erkennen. Modelle mit einer Auflösung von $voxsize = 0,1\ mm$ sind jedoch mit den implementierten Methoden nicht in Echtzeit zu berechnen. Sowohl beim Modell M_G als auch beim Modell M_Z liegen die Simulationszeiten t_{sim} dabei weit über $150\ ms$. Der verwendete Laser AT Fidelis arbeitet im Bereich von $f_p \in [10\ Hz; 50\ Hz]$. Zur Echtzeitberechnung und -visualisierung in diesem Takt wären Simulationszeiten von $t_{sim} \in [20\ ms; 100\ ms]$ nötig. Die Berechnungszeiten des Modells M_G wachsen mit steigender Kratertiefe h. Das implementierte Modell M_G ist daher vorerst nicht für eine Echtzeitanwendung, wie zur intraoperativen Abtragsberechnung, geeignet. Die Berechnungszeit beider Modelle steigt mit wachsender Gesamtanzahl N der Voxel des Laserobjekts. Bei großen Laserobjekten wie dem verwendeten Femur F_1 beträgt t_{calc} ca. $50\ ms$, während t_{calc} bei kleinen Objekten wie dem Würfel W_3 bei gleicher Auflösung weniger als $1\ ms$ beträgt. Beim Modell M_Z treten Fehler in der Kraterform auf, wenn für den Winkel zwischen dem Laserstrahl $\boldsymbol{n}$ und der bestrahlten Fläche $\beta \neq 90°$ gilt. Dieser Fehler ist unter der in Abschnitt 4.5 getroffenen Annahme, dass der Roboter den Laser immer senkrecht zur bestrahlten Fläche ausrichten kann, zu vernachlässigen. Der Fehler in der Kraterform des Modells M_G, welcher bei Winkeln $\beta \neq 90°$ zwischen dem Laserstrahl $\boldsymbol{n}$ und der Ausrichtung des Octreekoordinatensystems ${}^{oct}\boldsymbol{T}_{world}$ auftritt, ist dagegen nicht zu vernachlässigen, da der Bediener während der Planung und Simulation eines Abtrags kein Wissen über die Orientierung von ${}^{oct}\boldsymbol{T}_{world}$ hat. Das Modell M_G ist daher auch bei Simulationen, welche nicht in Echtzeit durchgeführt werden müssen, vorerst nur für den Sonderfall von $\beta = 90°$ anwendbar. Bei der Simulationsart S_{LP} mit Modell M_Z treten Hinterschneidungen im Voxelabtragsvolumen auf. Wie sich dieser Fehler bei mehrmaligem Überfahren der Trajektorie auf die Tiefenentwicklung des Abtragsvolumens auswirkt wurde nicht untersucht. Das Abtragsvolumen wurde bei dieser Simulationsart nicht vermessen, sondern nur qualitativ anhand der Form beurteilt.

5.2 Experiment zur Beurteilung des Arbeitsablaufs

In diesem Experiment wurde der Arbeitsablauf von der Planung der Laserosteotomie bis zur intraoperativen Abtragsberechnung getestet um Schwachstellen zu identifizieren. Als physisches Modell wurde dazu ein Femur aus Kunststoff verwendet. Von dem Knochen lagen keine CT-Daten vor, sondern ein Oberflächenmodell in Form einer stl-Datei. Daher konnte das Volumenmodell des entsprechenden Laserobjekts in der Planungsumgebung nicht, wie in Abschnitt 4.2 beschrieben ist, in Amira aus CT-Daten abgeleitet werden. Stattdessen wurde die stl-Datei mit einem am DLR entwickelten Füllalgorithmus [47] mit Voxeln gefüllt und eine vox-Datei (vgl. Anhang B) daraus erstellt. Diese konnte mit einem Parser in den Octree der Operationsplanung eingelesen werden, woraus die Volumenmodelle der Laserobjekte F_1 ($voxsize = 0,5\ mm$) und F_2 (der Größe $voxsize = 0,3\ mm$) entstanden. Da die Oberfläche der stl-Datei nicht geschlossen war, entstanden dabei große Hohlräume im Volumemmodell. Bestimmte Teile des Arbeitsablaufes konnten daher mit diesen Laserobjekten nicht gezeigt werden. Der Kunststofffemur und das Laserobjekt F_1 sind in Abbildung 5.13 zu sehen. Die Parameter der Laserobjekte F_1 und F_2 sind in Tabelle 5.1 zu finden.

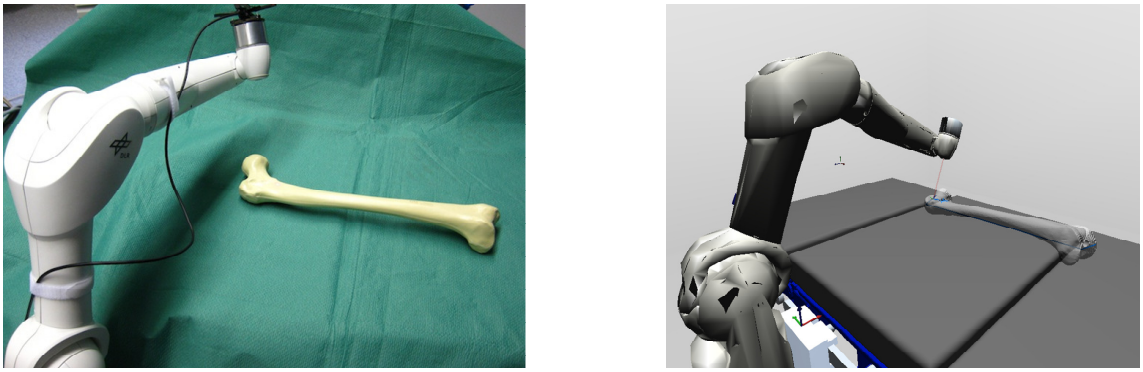


Abbildung 5.13: Aufbau des Experiments. Links: Der Kunststofffemur auf dem Operationstisch. Rechts: Das Laserobjekt F_1 in der simulierten Umgebung der Operationsplanung.

Zuerst wurde, wie in Abbildung 5.14 zu sehen ist, im Planungssystem jeweils eine der in Abschnitt 4.4 erklärten Trajektorienarten 1-4 eingezeichnet.

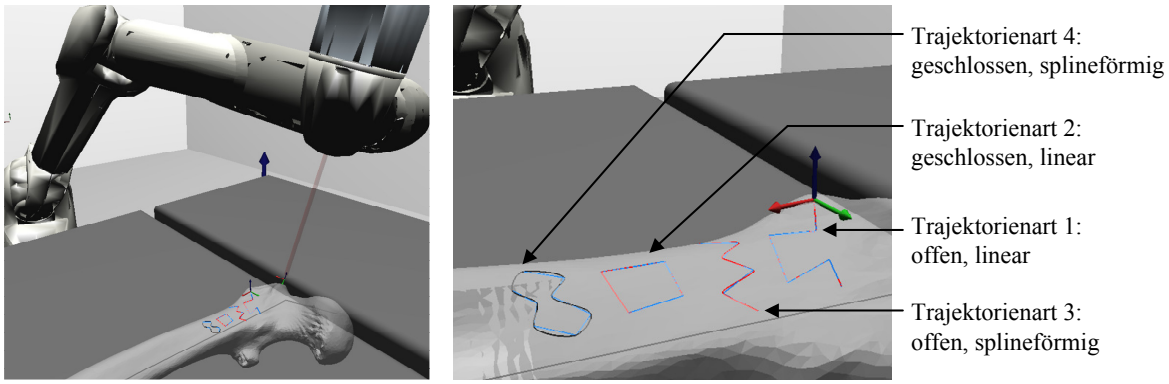


Abbildung 5.14: Verschiedene auf dem Femur eingezeichnete Trajektorien.

Für die offene, splineförmige Trajektorie wurde die Schnittnormale eingestellt und eine präoperative Simulation mit dem Laserobjekt F_1 durchgeführt. Dafür wurde die Simulationsart S_{LP} und das Modell M_Z ohne Berücksichtigung der Debris und mit einer Pulsfrequenz von $f_p = 20\ Hz$ sowie einer Pulsenergie von $P_E = 0,75\ J$ verwendet. Da das Volumenmodell von F_1 nur am Rand mit Voxeln besetzt war, wurde der in Abschnitt 4.4 erklärte Schritt Anzeigen des Abtragsvolumens nicht durchgeführt. Aus demselben Grund wurde nur eine Überfahrung simuliert, das Resultat ist in Abbildung 5.15 dargestellt.

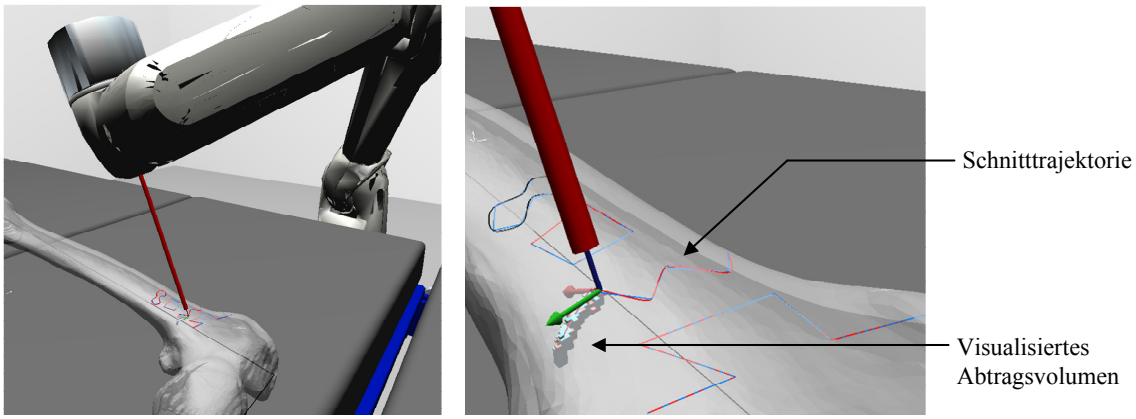


Abbildung 5.15: Präoperative Abtragssimulation und Visualisierung im Geoserver.

Durch die präoperative Abtragssimulation konnte sichergestellt werden, dass der Roboter die Trajektorie entlangfahren kann. Die Planungsdaten wurden exportiert und in die Robotersteuerung eingelesen. Mithilfe der Planungsdaten wurde eine Laserosteotomie an dem Kunststoffknochen nachgestellt, indem ein Laserpointer an den Operationsroboter MIRO angebracht wurde. Da im Rahmen dieser Arbeit keine Registrierungsmethode in den Arbeitsablauf implementiert wurde, musste der Femur manuell auf dem Operationstisch positioniert werden. Dafür wurden im Planungssystem einige charakteristische Punkte des Femurs markiert und vom Roboter angefahren. Dann wurde die Kommunikation zwischen Robotersteuerung und Planungssystem hergestellt und die Laserosteotomie mit einer intraoperativen Abtragsberechnung im Planungssystem durchgeführt. Es wurde zuerst F_1 und dann F_2 verwendet um festzustellen, wie sich der Unterschied der Voxelgrößen auf die intraoperative Abtragsberechnung auswirkt. In Abbildung 5.16 ist ein Matlab Plot der vom Roboter angefahrenen Punkte auf das Foto des Femurs gelegt. Es ist sichtbar, dass der Roboter die geplante Trajektorie entlangfahren konnte. Auf der rechten Seite sind die Visualisierungen der intraoperativen Abtragsberechnung mit F_1 und F_2 im Geoserver zu sehen. Bei der Simulation mit F_2 tritt ein Fehler auf. Die Berechnungs- und Visualisierungszeiten des Modells sind zu groß um alle von der Robotersteuerung gesendeten Datenpakete zu empfangen. Bei zeitlich weit auseinander liegenden Datenpaketen wird zwischen diesen interpoliert und die Pulse gemäß f_p und der Zeitdifferenz positioniert (vgl. Abschnitt 4.7). Daraus resultiert der sichtbare Wegunterschied zwischen Trajektorie und Abtragsvolumen.

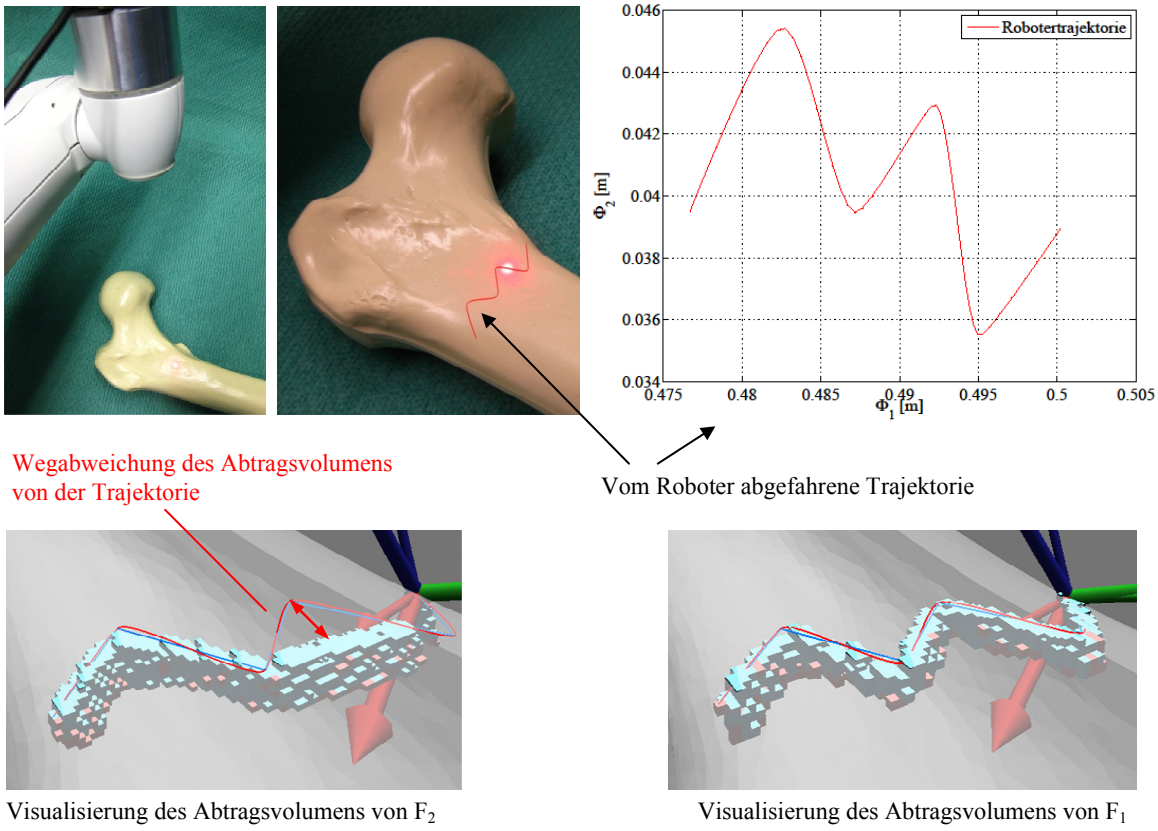


Abbildung 5.16: Durchführung der nachgestellten Laserosteotomie mit intraoperativer Abtragsberechnung und Visualisierung im Geoserver. Bei hohen Berechnungszeiten ergibt sich eine Wegabweichung des berechneten Abtragsvolumens.

Zusammenfassung Mit diesem Experiment konnte die Durchführbarkeit des Arbeitsablaufes gezeigt werden. Dabei ist die Planung von vier Trajektorienarten unterschiedlichen Komplexität, die präoperative Simulation und der Export der Daten möglich. Es wurde die Möglichkeit der Umsetzung der Planung anhand einer nachgestellten Laserosteotomie mit einem Kunststofffemur gezeigt. Der Roboter konnte die geplante Trajektorie entlangfahren. Dies wurde nur mit der offenen, splineförmigen Trajektorienform 3 gezeigt. Das in den Abschnitten 4.4 und 4.7 erklärte Einkoppeln am Startpunkt der Trajektorie mit $|\boldsymbol{v}| \neq 0$ konnte nicht gezeigt werden. Die parallel ablaufende Abtragssimulation innerhalb des Planungssystems war möglich. Dabei wurde deutlich, dass für eine echtzeitfähige interaoperative Simulation mit $f_p = 20\ Hz$ vorerst nur eine Voxelgröße von $voxsize = 0,5\ mm$ und das Modell M_Z verwendet werden können.

Kapitel 6

Diskussion und Ausblick

Im Rahmen dieser Arbeit wurde ein möglicher Arbeitsablauf für die Planung einer robotergestützten Laserosteotomie entwickelt und dazu ein prototypisches System implementiert. Das System ermöglicht es in einer simulierten Umgebung an dreidimensionalen anatomischen Strukturen Laserschnitte zu definieren. Die Laserosteotomie kann präoperativ simuliert und visualisiert werden. Die Simulation berücksichtigt die Roboterkinematik des am DLR entwickelten Leichtbauroboters MIRO, die Laserparameter des chirurgischen Er:YAG Lasers AT Fidelis und die laserspezifischen Materialparameter von Knochen. Mit der präoperativen Simulation kann geklärt werden, ob der Schnitt mit dem verwendeten Roboter durchführbar ist und wie sich der Abtrag entlang des Schnitts verhält. Nach Abschluss der Planung können die Parameter exportiert und für die Steuerung des Roboters und des Lasers verwendet werden. Intraoperativ kann eine Verbindung zwischen dem realem System und der simulierten Umgebung hergestellt sowie der Abtrag mit den realen Prozessparametern berechnet und visualisiert werden. Die Visualisierung ermöglicht dem Arzt während der Laserosteotomie Kontrolle über den Prozessfortschritt zu behalten. Zur Abtragsberechnung wurde der Knochen als Voxelvolumen modelliert, von welchem mit jedem Laserpuls eine bestimmte Menge an Abtragsvoxeln entfernt wird. Um auch größere Knochenstrukturen modellieren zu können wurde ein Octree verwendet. Während sich vergleichbare Systeme auf ca. 30.000 Voxel beschränken [56], konnten so Volumina mit bis zu $30\ Mio.$ Voxeln modelliert werden. Damit ist auch die Berechnung größerer Abtragsvolumina möglich. Es wurden zwei mathematische Abtragsmodelle implementiert. Eines berücksichtigt die gaußsche Energieverteilung im Strahlenquerschnitt und berechnet für jedes vom Laserstrahl getroffene Voxel den individuellen Energieverlust. Da die Voxel einzeln betrachtet werden, können Heterogenitäten wie verschiedene Knochendichten berücksichtigt werden. Bei Einwirken eines Pulses in homogenes Material ergibt sich eine gaußsche Kraterform, welche der in Experimenten gemessenen Kraterform [7] entspricht. Ein zweites Modell berechnet einen vereinfachten, zylindrischen Krater. Anstatt für jedes Voxel den spezifischen Energieverlust zu berechnen, werden im Zylinder liegende Voxel direkt entfernt. Das Modell ist stark vereinfacht und ermöglicht die Berücksichtigung von heterogenen Materialeigenschaften nicht. Die Schnitttiefe beider Modelle wurde anhand von Experimenten parametrisiert, welche in einer Vorgängerarbeit durchgeführt wurden [4], [19]. Die für den Knochenabtrag charakteristische Abnahme der Schnittgeschwindigkeit wurde dabei berücksichtigt. Es wurden Experimente durchgeführt um die Modelle bezüglich der Genauigkeit und der benötigten Rechenzeit zu vergleichen. Dabei wurde sichtbar, dass nur mit Voxelgrößen von ca. $0,1\ mm$ eine Formgenauigkeit zu erzielt

werden kann, mit der auch Einzelkrater in ihrer charakteristischen Form visualisiert werden können. Zur Abtragsberechnung bei dieser Voxelgröße waren mit beiden Modellen Rechenzeiten von $> 150\ ms$ nötig. Bei dem Modell zur Berechnung des gaußschen Kraters fiel außerdem auf, dass die Rechenzeiten mit wachsender Kratertiefe steigen. Daher war die intraoperative Abtragsberechnung vorerst nur mit Voxelgrößen $> 0,3\ mm$ und dem vereinfachten Modell möglich.
In Zukunft muss aber am ersten Modell weitergearbeitet und die Rechenzeiten müssen optimiert werden. Nur so können verschiedene Knochendichten berücksichtigt und eine hohe Allgemeingültigkeit des Modells erreicht werden. Möglich wäre z.B. die Parallelisierung von Berechnungen auf der Grafikkarte, wie es auch in anderen Bereichen der Simulation üblich ist [35]. Für eine bessere Adaption der Planung an die intraoperative Realität muss das System um eine Registrierungsmethode und eine Schnitttiefenmessung erweitert werden. Entsprechende Sensorik zur Schnitttiefenmessung wurde u.a. von Rupprecht [46] beschrieben. Außerdem sollte eine Evaluierung des Systems in Zusammenarbeit mit Ärzten geschehen, um das Erreichte mit den realen Anforderungen im klinischen Umfeld abzugleichen und Möglichkeiten der Weiterentwicklungen zu ermitteln. Da die Laserosteotomie bisher nur im Forschungsumfeld angewendet wurde, ist dabei die Art der Bedienung noch näher zu bestimmen. Im hier vorgestellten System wird der Schnitt offline in einer Simulationsumgebung definiert. Um besser auf intraoperative Änderungen eingehen zu können, wäre es denkbar das System um eine Onlinemethode zu erweitern. Für industrielle Laseranwendungen wurden Methoden entwickelt, bei denen zu lasernde Pfade direkt am Objekt durch ein getracktes Handgerät eingezeichnet werden und der entstehende Pfad durch Projektion auf das Objekt visualisiert wird [45]. Bezüglich der Bedienung muss auch der Modus geklärt werden, in dem der Roboter verwendet werden soll. Hier wurde davon ausgegangen, dass der Roboter die geplante Trajektorie autonom abfährt. Eine andere Möglichkeit ist die Telemanipulation des Roboters. Bei anderen Anwendungen mit dem Operationsroboter MIRO wird dies mit Force-Feedback Joysticks realisiert [60]. Bei Verwendung von solchen wäre es z.B. denkbar, die hier vorgestellte Abtragsberechnung zu verwenden, um dem Bediener beim telemanipulieren des Laserinstrumentes den haptischen Eindruck zu vermitteln, welcher beim direkten manipulieren eines Laserinstrumentes nicht entsteht.

Literaturverzeichnis

[1] AFILAL, S.: *Ablationsmechanismen von biologischem Hartgewebe bei Bestrahlung mit kurzgepulsten CO2-Lasern*, HHU Düsseldorf, Dissertation, 2004

[2] ALAVALA, C.R.: *CAD/CAM: Concepts and applications*. New Delhi, 2008

[3] ANDA, G.V.V. de ; ARNAUT, A.S. ; BEIZA, R.C. ; MACEDO, O. ; ESTRADA, N. ; ROMERO, L.: Robotics telepresence in acute care facilities during influenza AH1N1 outbreak in Mexico. In: *Critical Care* 14 (2010), Nr. Suppl 1, S. P475. – ISSN 1364-8535

[4] ARETZ, U.: *Materialsubstitution für Knochengewebe zur Erprobung von Schnittgeometrien und Ablationsmechanismen in der Laserosteotomie*, Fachhochschule Koblenz, Diplomarbeit, 2009

[5] BERLIEN, H.P. ; MÜLLER, G.J.: Applied laser medicine. In: *Journal of Biomedical Optics* 9 (2004), S. 844

[6] BODENMÜLLER, D.I.T.: *Streaming Surface Reconstruction from Real Time 3D Measurements*, Technische Universität München, Dissertation, 2009

[7] BURGNER, J.: *Robot assisted laser osteotomy*. KIT Scientific Publishing, 2010. – ISBN 3866444974

[8] BÄUML, B. ; HIRZINGER, G.: Agile robot development (aRD): A pragmatic approach to robotic software. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006

[9] BÖRNER, M. ; WIESEL, U.: Einsatz computerunterstützter Verfahren in der Unfallchirurgie. In: *Trauma und Berufskrankheit* 1 (1999), Nr. 2, S. 85–90. – ISSN 1436-6274

[10] CAMPBELL, C.J. ; KOESTER, C.J. ; CURTICE, V. ; NOYORI, K.S. ; RITTLER, M.C.: Clinical studies in laser photocoagulation. In: *Archives of Ophthalmology* 74 (1965), Nr. 1, S. 57

[11] CHARLTON, A. ; DICKINSON, M.R. ; KING, T.A. ; FREEMONT, A.J.: Erbium-YAG and holmium-YAG laser ablation of bone. In: *Lasers in Medical Science* 5 (1990), Nr. 4, S. 365–373. – ISSN 0268-8921

[12] CLEARY, K.: Medical robotics and the operating room of the future. In: *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the* IEEE (Veranst.), 2006, S. 7250–7253. – ISBN 0780387414

[13] CRAIG, J. J.: *Introduction to Robotics: Mechanics and Control*. Prentice Hall, 1986

[14] DUCK, F.A.: *Physical properties of tissue: a comprehensive reference book*. Academic Pr, 1990. – ISBN 0122228006

[15] EICHLER, J. ; EICHLER, H.J.: *Laser: Bauformen, Strahlführung, Anwendungen*. 2010. – ISBN 3642104614

[16] ENGELHARDT, A. ; BIMBERG, D.: Osteotomie mit Laser. In: *Laser und Elektro-Optik, Kapitel* 3 (1972), S. 54–57

[17] FOTONA: *Datenblatt AT Fidelis*. www.fotona.com: , März 2011

[18] GOTTSCHLING, H. ; ROTH, M. ; SCHWEIKARD, A. ; BURGKART, R.: Intraoperative, fluoroscopy-based planning for complex osteotomies of the proximal femur. In: *The International Journal of Medical Robotics and Computer Assisted Surgery* 1 (2005), Nr. 3, S. 67–73. – ISSN 1478-596X

[19] GRATZFELD, U.: *Evaluierung einer Tiefenkonstrolle für die Laserosteotomie*, Fachhochschule Koblenz, Diplomarbeit, 2009

[20] HAEGELEN, C. ; TOUZET, G. ; REYNS, N. ; MAURAGE, C.A. ; AYACHI, M. ; BLOND, S.: Stereotactic robot-guided biospies of brain stem lesions: Experience with 15 cases. In: *Neurochirurgie* (2010). – ISSN 0028-3770

[21] HAGN, U. ; KONIETSCHKE, R. ; TOBERGTE, A. ; NICKL, M. ; JÖG, S. ; KÜBLER, B. ; PASSIG, G. ; GRÖGER, M. ; FRÖHLICH, F. ; SEIBOLD, U. ; LE-TIEN, L. ; ALBU-SCHÄFFER, A. ; NOTHELFER, A. ; HACKER, F. ; GREBENSTEIN, M. ; HIRZINGER, G.: DLR MiroSurge: a versatile system for research in endoscopic telesurgery. In: *International journal of computer assisted radiology and surgery* 5 (2010), Nr. 2, S. 183–193. – ISSN 1861-6410

[22] HAGN, U. ; NICKL, M. ; JÖRG, S. ; TOBERGTE, A. ; KÜBLER, B. ; PASSIG, G. ; GRÖGER, M. ; FRÖHLICH, F. ; SEIBOLD, U. ; KONIETSCHKE, R. ; LE-TIEN, L. ; ALBU-SCHÄFFER, A. ; GREBENSTEIN, M. ; ORTMAIER, T. ; HIRZINGER, G.: DLR MIROSURGE–towards versatility in surgical robotics. In: *Proceedings of CURAC* 9 (2008), S. 143–146

[23] HAIDEGGER, T. ; SÁNDOR, J. ; BENYÓ, Z.: Surgery in space: the future of robotic telesurgery. In: *Surgical Endoscopy* (2010), S. 1–10. – ISSN 0930-2794

[24] HATWIG, J. ; REINHART, G. ; ZÄH, M.F.: Automated task planning for industrial robots and laser scanners for remote laser beam welding and cutting. In: *Production Engineering* 4 (2010), S. 1–6. – ISSN 0944-6524

[25] HIBST, R.: *Technik, Wirkungsweise und medizinische Anwendungen von Holmium-und Erbium-Lasern*. Ecomed-Verl.-Ges., 1997. – ISBN 3609511109

[26] HILLS, J.W. ; JENSEN, J.F.: Telepresence technology in medicine: principles and applications. In: *Proceedings of the IEEE* 86 (1998), Nr. 3, S. 569

[27] HOUNSFIELD, G.N.: Computed medical imaging. In: *Journal of Computer Assisted Tomography* 4 (1980), Nr. 5, S. 665. – ISSN 0363-8715

[28] IFR STATISTICAL DEPARTMENT: World Robotics 2010. (2010)

[29] KAHRS, L.A.: *Bildverarbeitungsunterstützte Laserknochenablation am humanen Felsenbein*. KIT Scientific Publishing, 2009. – ISBN 3866444583

[30] KLASING, M.: *Konstruktion und Evaluierung eines CO2-Laserosteotoms*, Fachhochschule Koblenz, Diplomarbeit, 2008

[31] KLODMANN, J.: *Kalibration des medizinischen Leichtbauroboters MIRO hinsichtlich der elastischen Roboterkinematik, der internen Momentensensorik und der Massenparameter*, Institut für Mechatronische Systeme, Universität Hannover, Diplomarbeit, 2010

[32] KONIETSCHKE, R.: *Planning of Workplaces with Multiple Kinematically Redundant Robots*, Technische Universität München, Dissertation

[33] KWOH, Y.S.: A robot with improved absolute positioning accuracy for CT guided stereotactic brain surgery. In: *Biomedical Engineering, IEEE Transactions on* 35 (1988), S. 153

[34] LAUTERBUR, P.C.: Medical imaging by nuclear magnetic resonance zeugmatography. In: *Nuclear Science, IEEE Transactions on* 26 (1979), Nr. 2, S. 2807–2811. – ISSN 0018-9499

[35] LEE, A. ; YAU, C. ; GILES, M.B. ; DOUCET, A. ; HOLMES, C.C.: On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods. In: *Journal of Computational and Graphical Statistics* 19 (2010), Nr. 4, S. 769–789. – ISSN 1061-8600

[36] LITHWICK, N.H. ; HEALY, M.K. ; COHEN, J.: Micro-Analysis of bone by laser microprobe. In: *Surgical forum* Bd. 15, 1964, S. 439

[37] MAIMAN, T.H.: Stimulated optical radiation in ruby. In: *Nature* 187 (1960), S. 493–494

[38] MITRA, T.: *Ablation biologischen Hartgewebes mit gepulsten IR-Lasern*, HHU Düsseldorf, Dissertation, 2002

[39] MOERIKE, K.D. ; BETZ, E. ; MERGENTHALER, W.: *Biologie des Menschen*. Quelle & Meyer, 1989. – ISBN 3494011532

[40] MORNEBURG, H.: *Bildgebende Systeme für die medizinische Diagnostik*. Publicis MCD-Verl., 1995. – ISBN 3895780022

[41] MÖNNICH, H. ; STEIN, D. ; RACZKOWSKY, J. ; WÖRN, H.: System for Laser Osteotomy in Surgery with the Kuka Lightweight Robot–First Experimental Results. In: *World Congress on Medical Physics and Biomedical Engineering, September 7-12, 2009, Munich, Germany* Springer (Veranst.), 2009, S. 179–182

[42] NETTER, F.H.: *Atlas der Anatomie*. Churchill Livingstone, 2008. – ISBN 3437416022

[43] ORTMAIER, T. ; GRÖGER, M. ; BÖHM, D.H. ; FALK, V. ; HIRZINGER, G.: Motion estimation in beating heart surgery. In: *Biomedical Engineering, IEEE Transactions on* 52 (2005), Nr. 10, S. 1729–1740. – ISSN 0018-9294

[44] PARKES, S.: Space Wire: the standard. In: *EUROPEAN SPACE AGENCY-PUBLICATIONS-ESA SP* 447 (1999), S. 111–116. – ISSN 0379-6566

[45] REINHART, G. ; MUNZERT, U. ; VOGL, W.: A programming system for robot-based remote-laser-welding with conventional optics. In: *CIRP Annals-Manufacturing Technology* 57 (2008), Nr. 1, S. 37–40. – ISSN 0007-8506

[46] RUPPRECHT, S. ; TANGERMANN, K. ; KESSLER, P. ; NEUKAM, F.W. ; WILTFANG, J.: Er: YAG laser osteotomy directed by sensor controlled systems. In: *Journal of Cranio-Maxillofacial Surgery* 31 (2003), Nr. 6, S. 337–342. – ISSN 1010-5182

[47] SAGARDIA, M.: *Enhancements of the voxmap-pointshell algorithm*, Universität Navarra, Diplomarbeit, 2008

[48] SCHAWLOW, A.L. ; TOWNES, C.H.: Infrared and optical masers. In: *Physical Review* 112 (1958), Nr. 6, S. 1940

[49] SCHMITZ-RODE, T.: *Runder Tisch Medizintechnik: Wege zur beschleunigten Zulassung und Erstattung innovativer Medizinprodukte*. Springer, 2009. – ISBN 3642025994

[50] SCHULZ A.P., Seide K.: Results of total hip replacement using the Robodoc surgical assistant system: clinical outcome and evaluation of complications for 97 procedures. In: *The international journal of medical robotics and couputer assisted surgery* 3 (2007), S. 301 – 306

[51] SHI, J. ; STENZEL, R. ; WENGER, T. ; LUETH, T.C.: Accuracy study of a new assistance system under the application of Navigated Control® for manual milling on a head phantom. In: *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE* IEEE (Veranst.), 2010, S. 2296–2299. – ISSN 1557-170X

[52] SHOHAM, M. ; BURMAN, M. ; ZEHAVI, E. ; JOSKOWICZ, L. ; BATKILIN, E. ; KUNICHER, Y.: Bone-mounted miniature robot for surgical procedures: concept and clinical applications. In: *Robotics and Automation, IEEE Transactions on* 19 (2003), Nr. 5, S. 893–901. – ISSN 1042-296X

[53] SPETZ, J.: Physicians and physicists: the interdisciplinary introduction of the laser to medicine. In: *Sources of medical technology: universities and industry* 5 (1995), S. 41. ISBN 0309051894

[54] STATISTISCHES BUNDESAMT: Operationen und Prozeduren der vollstationären Patientinnen und Patienten der Krankenhäuser 2009. (2009)

[55] STEIN, D. ; MONNICH, H. ; RACZKOWSKY, J. ; WORN, H.: Automatic and hand guided self-registration between a robot and an optical tracking system. In: *Advanced Robotics, 2009. ICAR 2009. International Conference on* IEEE (Veranst.), 2009, S. 1–5

[56] STOPP, S.: *Ein integriertes System für die Mund-, Kiefer- und Gesichtschirurgie und Implantologie*, TU München, Dissertation, 2008

[57] STRAUSS, G. ; SPITZER, C. ; DITTRICH, E. ; HOFER, M. ; STRAUSS, M. ; LÜTH, T.: Ein modifiziertes Verfahren zur bissschienenbasierten Patientenregistrierung für die HNO-Navigation. In: *HNO* 57 (2009), Nr. 2, S. 153–159. – ISSN 0017-6192

[58] TAYLOR, R.H.: Medical Robotics and Computer-Integrated Surgery. In: *Computer Software and Applications, 2008. COMPSAC '08. 32nd Annual IEEE International*, 282008-aug.1 2008, S. 1. – ISSN 0730-3157

[59] TAYLOR, R.H. ; STOIANOVICI, D.: Medical robotics in computer-integrated surgery. In: *IEEE transactions on robotics and automation* 19 (2003), Nr. 5, S. 765–781. – ISSN 1042-296X

[60] TOBERGTE, A. ; PASSIG, G. ; KUEBLER, B. ; SEIBOLD, U. ; HAGN, U.A. ; FRÖHLICH, F.A. ; KONIETSCHKE, R. ; JÖRG, S. ; NICKL, M. ; THIELMANN, S. ; HIRZINGER, G.: MiroSurge-Advanced User Interaction Modalities in Minimally Invasive Robotic Surgery. In: *PRESENCE: Teleoperators and Virtual Environments* 19 (2010), Nr. 5, S. 400–414. – ISSN 1054-7460

[61] TZAFESTAS, S.G.: *Robotic systems: advanced techniques and applications*. Springer, 1992. – ISBN 0792317491

[62] VERSCHUEREN, RC ; KOUDSTAAL, J. ; OLDHOFF, J.: The carbon dioxide laser; some possibilities in surgery. In: *Acta chirurgica Belgica* 74 (1975), Nr. 2, S. 197. – ISSN 0001-5458

[63] WIRTZ, D.C.: *Ae-Manual Der Endoprothetik: Knie*. Springer, 2010. – ISBN 3642128882

[64] WÖRN, H. ; PETERS, H. ; IVANENKO, M. ; HERING, P.: LASER Based Osteotomy with Surgical Robots. In: *Biomedizinische Technik* 50 (2005), S. 25–6

[65] ZARET, MM ; RIPPS, H. ; SIEGEL, IM ; BREININ, GM: Laser photocoagulation of the eye. In: *Journal of Occupational and Environmental Medicine* 5 (1963), Nr. 8, S. 408. – ISSN 1076-2752

[66] ZÄH, M.F. ; HATWIG, J. ; MUSIOL, J. ; RÖSCH, O. ; REINHART, G.: Analysis of the Accuracy of Industrial Robots and Laser Scanners for Remote Laser Beam Welding and Cutting. In: *Proceedings-ISR/ROBOTIK 2010* (2010)

[67] ZÄH, MF ; MOESL, J. ; MUSIOL, J. ; OEFELE, F.: Material processing with remote technology revolution or evolution? In: *Physics Procedia* 5 (2010), S. 19–33. – ISSN 1875-3892

[68] ÖMÜRLÜ, Y.: *Bedeutung der zweidimensionalen Operationsplanung für Hüfttotalendoprothesen*, Ruhr-Universität Bochum, Dissertation, 2005

Anhang

Anhang A

Softwaredokumentation

Die GUIs für die Benutzerschnittstelle wurden in Qt programmiert, das Berechungsmodul in C++. Aufgrund des Umfangs des Quellcodes werden nur die wichtigsten Klassen mit ihren Membervariablen und Funktionen aufgelistet. Da die Kommentare direkt aus dem Quellcode gewonnen wurden, ist die Dokumentation auf Englisch.

- **oppstep1impl.cpp**: Diese Klasse repräsentiert die Lasersetting GUI
- **VolObject.cpp**: Diese Klasse repräsentiert ein einzelnes Laserobjekt
- **VolObjects.cpp**: Diese Klasse repräsentiert eine Gruppe von Laserobjekten
- **robot.cpp**: Diese Klasse repräsentiert einen Roboter
- **Lasertool.cpp**: Diese Klasse repräsentiert ein Laserwerkzeug

Mat4 entspricht einer homogenen Matrix, **Vec3** einem Vektor, alle Datentypen im Namespace **L3D** kommen aus der am DLR entwickelten L3D-Bibliothek, welche auch den Octree enthält.

opp_step1Impl Class Reference

Public Member Functions

- void updateLaserobjectsList ()

  To update List of Laserobjects (which are only the VolObjects in thisOpp->vo) Only done at the beginning in **initform()**.

- void updateLinetypeList ()

  To update if List get's interpolated spline or linear, closed or open Stored in Qt Class combo-box, no member Only done at the beginning in **initform()**.

- void manageLaserGuiElements ()

Checks if... **button_pickLaserline** *(QRadioButton) ("Pick laserline")* **button_CreateInterpolline** *(QPushButton) ("Create Trajectory")* **button_Undo** *(QPushButton) ("Undo")* **button_ShowAblationslice** *(QPushButton) ("Create Ablationslice")* **button_SingleshotSimulation** *(QPushButton) ("Start Simulation") ...can be set Only done at the beginning in* **initform()** *and in* **LaserLinenameSlot**.

- void checkPickLaserline ()

  Enables geoserver functions to catch points and draw a line out of them Only possible if **button_pickLaserline** *is set, happening in* **manageLaserGuiElements()**.

- void setAblNormal ()

  To show ablationdirection with Coordinatesystem in geoserver, depending on **Pitch, ...** *angles Done at the end of* **CreateInterpollineSlot()** *and at* **SpinboxAbldirPitchSlot** *usw.*

- void pickLaserlineSlot ()

  check if QtElements can get enabled via **manageLaserGuiElements()** *when* **manageLaserGuiElements()** *is changed*

- void LaserobjectsSlot (const QString &)

  Check the buttons via **manageLaserGuiElements()** *when new Laserobject is chosen.*

- void LaserLinenameSlot (const QString &)

  Choose an existing line or create a new one in **combobox_Linename** *if new one is created first has to be saved with* **CreateInterpollineSlot()**.

- void UndoSlot ()

  Removes the last set point, with the **UNDO button** *and geoserverfunction* **sendRemovePoint**.

- void CreateInterpollineSlot ()

  Save the chosen line and create and save a interpolated line for it (with name "linename_interp") depending on which linetype had been chosen, save them both as independent **line** *a normal gets assigned which is not saved yet at this point and only appears in* **spinbox_AblationdirectionX** *and* **spinbox_AblationdirectionPitch</bb>.**

- void SpinboxAbldirPitchSlot (double d_newval_pitch)

  Make change in Ablationdirection pitch angle possible and consider the other angles and normaldirection boxes.

- void SpinboxAbldirYawSlot (double d_newval_yaw)

  Make change in Ablationdirection yaw angle possible and consider the other angles and normaldirection boxes.

- void SpinboxAbldirRollSlot (double d_newval_roll)

Make change in Ablationdirection Roll angle possible and consider the other angles and normaldirection boxes.

- void ShowAblationsliceSlot ()

  Create an Ablationslice with all it's **_laserobjects.at(i)._volume** *- functions.*

- void ExportTrajectorySlot ()

  Exports the Robotposes along the line with respect to the ablationdirection into a Matlab readable Format.

- void SingleshotSimulationSlot ()

  Start a Simulation, where only one shot (middlepoint of traject) gets lasered (once with every button click).

- void LineshotSimulationSlot ()

  Start a Simulation, where the whole traject gets lasered (once with every button click).

- void CalculateSlot ()

  Calculates computing time of singleshots.

VolObject Class Reference

Public Member Functions

- VolObject ()

  Default constructor. Creates an empty Laserobject.

- ∼VolObject ()

  Default destructor.

- VolObject (const VolObject &src)

  Copy constructor. Makes a deep copy of **src**.

- VolObject & operator= (const VolObject &src)

  Assignment operator. Makes a deep copy of **src**.

- void getDatafromAmiraFile (std::ifstream &sourcefile)

  To get data from a Amira processed .am-file (ASCII Amira Mesh) and fill octree with it.

- void getDatafromVoxFile (std::ifstream &sourcefile)

To get data from a VpsGenerator processed .vox file and fill octree with it.

- void ShowAblationVolume (Line ∗interpolline, Vec3 ablnormal, float width)

  To show the Ablationslice defined by normal, trajekt and width in the geoserver, by creating a VoxelObject and writing single Voxel in it with sendsetVoxel().

- void exportAblationVoxel (std::string filename, unsigned char state, bool halfablated)

  Export current ablation voxel to file to analyse them.

- void calculatePathAblation (Mat4 lastTCP, Mat4 currTCP, unsigned int pulsenum, float intensity, const Lasertool &lasertool, bool depthincrease)

  To interpolate between lastTCP and currTCP and set **pulsenum** *pulses with calculatePointAblation2().*

- void calculatePointAblation2 (L3D::Beam beam, const Lasertool &lasertool, std::string simtype, bool depthincrease)

  To calculate pulse depending on Simulationtype **simpyte***, which can be cylinder or gauss.*

- void calculatePointAblationForTimeCalculation (L3D::Beam beam, const Lasertool &lasertool, std::string simtype, bool depthincrease)

  Same as calculatePathAblation(), but for measuring calculationtime.

- void findPossibleVoxelCylinder (L3D::Beam beam, L3D::Map< float, L3D::DistanceDataCube< L3D::SpaceState > > &volMap, const Lasertool &lasertool, bool depthincrease)

  To calculate Ablation with the model cylinder, using getAblationDepth().

- void findPossibleVoxelGauss (L3D::Beam beam, L3D::Map< float, L3D::DistanceDataCube< L3D::SpaceState > > &volMap, const Lasertool &lasertool, bool depthincrease)

  To calculate Ablation with the model gauss, using getAblationDepth() and calculateVoxelEnergyLoss().

- float getAblationDepth (float h, float r, float w, float PE, bool depthdependency)

  Get the maximal depth with these laserparameters.

- void calculateVoxelEnergyLoss (L3D::Beam beam, L3D::Map< float, L3D::DistanceDataCube< L3D::SpaceState > > &volMap, const Lasertool &lasertool, bool depthincrease)

  To calculate each voxels energy loss, uses in findPossibleVoxelGauss().

- std::string getName ()

  get Name of Laserobject

- void sendShowVolObject2 ()

  Show laserobject in geoserver.

- void sendHideVolObject2 ()

  Hide laserobject in geoserver.

- void sendLoadVolObject2 () const

  Load laserobject into geoserver.

- void sendDeleteVolObject2 ()

  Delete laserobject in geoserver.

- void sendSetVolOpaqueness2 (double d)

  Set Opacity of laserobject in geoserver.

- int setVolObject (FILE ∗fp, char ∗name)

  Get infos from .def-file and load them into geoserver.

- void sendLoadVoxelObject2 () const

  Load Voxelobject of laserobject into geoserver.

- void sendDeleteVoxelObject2 ()

  Delete Voxelobject of laserobject in geoserver.

- void sendSetVoxel2 (unsigned long long ID, L3D::fPoint3 base, float size, unsigned char EN_new, unsigned char EN_old)

  Load single voxel of voxelobject of laserobject into geoserver.

Public Attributes

- float _translboundingbox [3]

  offset for octree use (can only start at (0,0,0))

- float _boundingbox [2][3]

  first (lower left front) and last (upper right back) point. x,y,z coordinates

- float _voxsize

  same size in all three directions in cm

- int _vox_amount [3]

  amount of voxel in each direction

- int _voxel_num

  Amount of all Voxel.

- L3D::SimpleSpace< L3D::SpaceState > _voxel

  octree for whole bone

- bool _octree_filled

  to see if octree had been filled with .am-File

- usecase _use

  To define the use of VolObject. Can be 0:NORMAL_VOLUME or 1:ABL_VOLUME.

- std::string _name

  name of laserobject

- std::string _path

  path of laserobject

- std::string _relpath

  relpath of laserobject

- Mat4 _pos

  aposition and rotation as written in the .def-file is the Trafomatrix objectcoords–>virtcoords

- double _scale

  scale (as written in the .def-file) to scale into meters

- int _visible

  visibility as written in the .def-file

- int _collobj

  Collision Object

- double _opaque

  Opaqueness as written in the .def-file.

- int _movable [6]

  If Object is movable in 6DoF as written in .def-file.

- float _alpha

  Materialparameter: Absorptioncoefficient alpha in [cm^(-1)].

- std::map< float, float > _mu_h

  Materialparameter: Abschwächungskoeffizient mu(h), map key ist h [cm]! (Abschwächung durch Ablationsprodukte) mu in [cm^(-1)].

- float _Phi_s

  Materialparameter: Ablationsschwelle (ab dieser Energiedichte geschieht erst Ablation) in [J/cm^2].

- float _h

  Schnitt- oder Kratertiefe [cm].

VolObjects Class Reference

Public Member Functions

- VolObjects ()

  Default constructor. Creates an empty vector of Laserobject.

- ~VolObjects ()

  Destructor.

- VolObjects (const VolObjects &r)

  Copy constructor. Makes a deep copy of **src**.

- VolObjects & operator= (const VolObjects &r)

  Assignment operator. Makes a deep copy of **src**.

- int getVolObjectNum (std::string name)

  Get number of laserobject specified by name.

- void addVolObject (VolObject &vo)

  Add laserobject to vector.

- int getNumVolObjects ()

  Get amount of all laserobjects.

- void setVolObject (int &i, VolObject &vo)

  Set specific laserobject.

- VolObject & getVolObject (int &i)

  Get laserobject specified by number in vector.

Public Attributes

- vector< VolObject > _volobjects

  Vector of laserobjects.

Robot Class Reference

Public Member Functions

- void saveTrajectoryForMatlab (Vec3 ablnormal, Line ∗line, std::string vo_name, Mat4 vo_pos, bool STRAIGHT_SPLINE, bool OPEN_CLOSE, int frequency)

  Export planning parameters to Matlab .m-File.

- int sendSetRobot2 (Mat4 T_R_wanted)

Public Attributes

- std::vector< Dhpar > _dhpar

  dh parameters

- std::vector< Part > _part

  joints and other parts

- Mat4 _base

  Robotbase.

- Mat4 **_lastJointToTcp**
- std::vector< Task > _task

  vector of different tasks

- char _name [128]

  name of robot

- char _basename [64]

  base to which robot is attached

- int _baseIndex

  index of object or robot to which robot is attached

- int _visible

  Visibility in geoserver.

- int _dhmode

  craig = 1

- double _slidescale

  factor for GUI translational slider

- int **_actMount**
- std::vector< int > **_slidejoint**
- std::vector< double > _phi0

  initial angles

- std::vector< std::vector< double > > **_phi0mount**
- std::vector< double > _phimin

  lower joint limit

- std::vector< double > _phimax

  upper joint limit

- std::vector< double > _velmax

  maximum velocity of joint i

- std::vector< double > _enc

  encoder resolution of joint i

- Robottype **_rType**
- Lasertool _lasertool

  Lasertool adapted to TCP of robot.

- int **_actTask**

Lasertool Class Reference

Public Member Functions

- Lasertool ()

  Default constructor. Creates an empty Lasertool.

- ∼Lasertool ()

  Default destructor.

- Lasertool (const Lasertool &src)

  Copy constructor. Makes a deep copy of **src**.

- Lasertool & operator= (const Lasertool &src)

  Assignment operator. Makes a deep copy of **src**.

Public Attributes

- L3D::Beam _laserbeam

  Laserbeam consisting of beambase and direction.

- float _z_r

  Rayleighlength [mm].

- float _w_0

  Laserbeamdiameter [mm].

- float _z

  Distance from focus [mm].

- float _PE

  Pulseenergy [mJ].

Anhang B

Verwendete Dateiformate

Die wichtigsten verwendeten Dateiformate werden hier kurz erklärt:

lin-Datei

Eine lin-Datei beschreibt eine Linie im Geoserver, wie sie für die Darstellung der Robotertrajektorien verwendet wird. Eine selbsterklärende Beispieldatei mit einer aus vier Punkten bestehenden Trajektorie ist unten dargestellt.

```
max points, num points, closed, height
1000 4 0 0.000000
shine, red, green, blue
30.000000 0.000000 0.500000 1.000000
points, normals
-0.548017 0.790670 -0.096516 -0.167679 0.881764 -0.440882 -0.534299 -0.457120 -0.711032
-0.828498 0.116338 0.547775
-0.552580 0.791308 -0.089558 -0.134864 0.966611 -0.217888 -0.191114 -0.241143 -0.951486
-0.972259 -0.086680 0.217254
-0.561201 0.791619 -0.094767 0.377711 0.902347 -0.207618 -0.095745 -0.184963 -0.978070
-0.920960 0.389306 0.016532
-0.564012 0.794168 -0.091837 0.661985 0.722174 -0.200601 -0.097507 -0.182388 -0.978380
-0.743148 0.667233 -0.050321
```

obj-Datei

Die obj-Datei beschreibt ein 3D Objekt im Geoserver. Die dargestellte Datei beschreibt ein rotes Objekt, bestehend aus sechs Ebenen (f für faces), welche durch acht Ecken (v für vertrices), und sechs Normalen (vn für normal) besteht.

```
newmtl red
        Ns 40
        d 1
        illum 2
        Kd 0.698039 0.133333 0.133333
        Ks 0.698039 0.133333 0.133333
        Ka 0.698039 0.133333 0.133333

v  2.532071 -10.627805 -38.005001
v  2.930808 -11.004375 -37.999428
v  2.008955 -10.997607 -38.001362
v  2.532071 -10.627805 -38.005001
v  3.001718 -10.931617 -37.997627
v  2.930808 -11.004375 -37.999428
v  1.998232 -10.000712 -38.001842
v  2.532071 -10.627805 -38.005001

vn 0.002005 -0.012676 -0.999918
vn 0.019444 0.005792 -0.999794
vn -0.006566 -0.000553 -0.999978
vn 0.013115 -0.003995 -0.999906
vn -0.001216 0.004002 -0.999991
vn 0.022223 -0.011548 -0.999686
vn -0.999547 -0.010765 -0.028114

usemtl red
f 1/0/1 2/0/1 3/0/1
f 4/0/2 5/0/2 6/0/2
f 7/0/3 8/0/3 9/0/3
f 10/0/4 11/0/4 12/0/4
f 13/0/5 14/0/5 15/0/5
f 16/0/6 17/0/6 18/0/6
```

am-Datei

In Amira bearbeitete CT- oder MRT-Daten können in Form einer am-Datei als Voxelvolumen exportiert werden. Die unten dargestellte Datei beschreibt einen Wirbel, dessen Voxel zwei Zustände haben können (Bone und Exterior). Er ist definiert durch die Maße seiner Boundingbox und die Voxelmengen in x-,y- und z-Richtung. Ab der Zeile *Data section follows* werden alle Voxel mit deren Zustand aufgelistet. Die Liste beschreibt in diesem Fall $82 \times 87 \times 75$ Voxel, welche zuerst in x-, dann in y- und dann in z-Richtung aufgelistet sind. Die Datei ist hier nur bis zum zehnten Voxel dargestellt.

```
# AmiraMesh 3D ASCII 2.0

# CreationDate: Mon Dec 13 13:15:57 2010


define Lattice 82 87 75

Parameters {
    Materials {
        Exterior {
            Id 1
        }
        Bone {
            Id 2
        }
    }
    ImageData "Model_EinWirbel_ASCII.am",
    Content "82x87x75 byte, uniform coordinates",
    BoundingBox 0.624511 4.67451 -13.0801 -8.78008 -39.675 -35.975,
    CoordType "uniform"
}

Lattice { byte Labels } @1

# Data section follows
@1
0
0
0
0
0
1
1
1
1
0
```

vox-Datei

Eine vox-Datei beschreibt ein DLR internes Voxelformat. Mit dem Programm VpsGenerator können damit aus stl-Dateien Voxeldateien erzeugt werden. Der Header der (nicht vollständig) dargestellten Datei ist selbsterklärend.

```
# VoxelMap generated by vpsGenerator V0.97 on Wed Mar 16 10:42:08 2011
# Command: vpsGenerator -i
/volume/USERSTORE/lohm_ma/2_AmiraData/110301_Femur/amira-surface_afnt.iv -s 0.1 -l -10
10 1 -d 1 -b 255 255 255
# Directory: /volume/USERSTORE/lohm_ma/Voxeliser/vps_Mikel/VpsGenerator
# Time needed for generation: 15.9823sec
# Original iv-file: /volume/USERSTORE/lohm_ma/2_AmiraData/110301_Femur/amira-
surface_afnt.iv
#
# An VoxMap-inputfile looks like this:
# 2                 Type: 1=Standard, 2=Simple Compression, 3=Binary
# -29.0 -7.75 -16.5   origin or center of mass of voxel model
# 13 2 7              number of voxels
# 4.0               size of voxel
# -4 6              min and max value for voxels
# 1500                number of voxels
# 182               number of compressed entries
# 4  0              combined 4 Voxels with same value - normalized value
# 2  1              combined 2 Voxels with same value - normalized value
# 5  2              combined 5 Voxels with same value - normalized value
# ...               ...
# With these entries, the voxelmap would be: [-4,-4,-4,-4,-3,-3,-2,-2,-2,-2,-2, ...]
#
# The values may be:
# <CODE> < 0 </CODE> : empty space
# <CODE> = 0 </CODE> : border of an object
# <CODE> > 0 </CODE> : inside an object
# ---
2
-7.64392 -5.60882 -1.10254
142 95 526
0.1
-10 10
7095740
164168

75707 0
2 1
523 0
7 1
```

m-Datei

Die m-Datei wird verwendet um die Planungsdaten zu exportieren und in das Simulinkmodell, mit welchem der Roboter angesteuert wird, einlesen zu können. Zuerst werden die Laserparameter Pulsenergie P_E und Frequenz f_p, dann die Roboterbasis ${}^{W}\mathbf{T_{RB}}$ und ie DH-Parameter exportiert. In dieser Datei wird noch eine Registrierungslinie zur Positionierung des Laserobjekts angegeben. Danach werden die Trajektorienparameter Trajektorienart, $\dot{\Theta}_{i,max}$ mit $i \in \{0; 1; 2\}$ und P_j und die Interpolatorparameter exportiert.

```
%Planned Laserosteotomie TUE, 11.35 a.m.
%
%Laser: AT Fidelis
PE=0.75;        %[J]
fp=20;      %[Hz]
%
%Robot: miro_Inst1
T_W_RB=[1.83691e-16 -1 -1.22461e-16 -0.509338 ;
-3.06152e-16 -1.22461e-16 1 0.5831 ;
-1 -1.83691e-16 -3.06152e-16 0.3926 ;
0 0 0 1 ];  %[m]
dhParametersKineMedicOnly(7,3)=-0.13;       %[m]
%Laserobject: block0_01
RegLinePoses=[0.0000   -0.0000   -1.0000    0.5387  -1.0000   -0.0000   -0.0000↙
0.1163  -0.0000    1.0000   -0.0000    0.2000  0 0 0 10.0000   -0.0000   -1.0000↙
0.4859   -1.0000   -0.0000   -0.0000    0.1085  -0.0000    1.0000   -0.0000    0.1893  0↙
0 0 10.0000   -0.0000   -1.0000    0.4616   -1.0000   -0.0000   -0.0000   -0.3198↙
-0.0000    1.0000   -0.0000    0.2079  0 0 0 10.0000   -0.0000   -1.0000    0.5132↙
-1.0000   -0.0000   -0.0000   -0.3301   0.0000    1.0000   -0.0000    0.2025  0 0 0 1];↙
%[m]
%
%Trajectory: /home/lohm_ma/workspace/data/tmp/blockline2_0_01_interp
STRAIGHT_SPLINE = 1;
OPEN_CLOSED = 0;
acc_loops = 0;
v_start = [0.1534 0.0716 0.1044];
ISOPoses=[0.984 -0.178169 2.79382e-16 0.177748 0.981677 0.0686708 -0.012235 -0.067572↙
0.997639 0.480574 -0.054663 0.181959 0.984 -0.178169 ...
%
%Interpolator
params=[0.2,2];
nrOfSteps= 48;
```

Abbildungsverzeichnis

1.1 Mögliche Schnittformen bei der Laserosteotomie 1

2.1 Möglicher Aufbau einer Laserosteotomie. 3
2.2 Anatomischer Aufbau eines Femurknochens. 4
2.3 Grundaufbau eines Lasers. 6
2.4 Grundprinzip eines Lasers. 7
2.5 TEM Moden eines Lasers. 8
2.6 Hartgewebe - Laser Wechselwirkungsmechanismen. 9
2.7 Remote Laserstrahlschweißen. 10
2.8 Der Operationsroboter DaVinci von Intuitive Surgical. 12
2.9 Das System MiroSurge, entwickelt am DLR. 13
2.10 Robotergestütztes Laserosteotomiesystem mit einem Stäubli RX90B CR Roboter. 14
2.11 Intraoperative Laserabtragsregelung durch Navigation. 15
2.12 Planungsmethoden mit 2D Röntgenaufnahmen 16
2.13 2D Voxelmodell nach Kahrs 18
2.14 3D Voxelmodell nach Stopp 18
2.15 Planungsmethode für eine Laserosteotomie nach Burgner 19

4.1 Einordnung des Planungssystems in das Operationsszenario. 24
4.2 Interner Aufbau des Planungssystems. 24
4.3 GUIs zur Realisierung der Benutzerschnittstelle. 25
4.4 Geometrische Auslegung des Operationsroboter MIRO. 27
4.5 Mögliche Rechnerstruktur für eine Einarmapplikation mit dem MIRO. 28
4.6 Der chirurgische Laser AT Fidelis von Fotona. 29
4.7 Arbeitsablauf von der Planung der Laserosteotomie bis zur intraoperativen Abtragsberechnung 30
4.8 Funktionsprinzip der Computertomographie. 31

4.9 Arbeitsablauf zur Vorbereitung der 3D Daten. 33
4.10 Räumlicher Aufbau der virtuellen OP Umgebung. 34
4.11 Einzelschritte zur Schnittplanung über die Bedienerschnittstelle. 35
4.12 Verschiedene Arten der Voumenmodellierung 36
4.13 Aufbau eines Octrees 37
4.14 Interpolation über Stützpunkte. 40
4.15 Implemtierung des Interpolators. 43
4.16 Erzeugung einer geschlossenen Linie mit dem Interpolator. 44
4.17 Profil eines Laserkraters. 45
4.18 Verschiedene Strategien zum Lasern eines Schnitts. 46
4.19 Vereinfachung des gaußschen Kraters zu einem kegelförmigen Krater. 49
4.20 Ergebnisse eines Experimentes zur Tiefenentwicklung. 50
4.21 Erste drei Schritte des Einzelpulsmodells M_G. 52
4.22 Zwischenschritt des Modells M_G: Definition eines Kubus, der den Zylinder umschließt. 54
4.23 Der Gesundheitszustand H. 55
4.24 Flussdiagramm zur Berechnung des neuen Gesundheitszustands H_{neu}. 56
4.25 Problem des Modells M_G bei mehreren Pulsen. 57
4.26 Bilden des Zylinders beim Modell M_Z. 58
4.27 Zuordnung der Voxel zum Zylinder beim Modell M_Z. 59
4.28 Einzelschritte zur Durchführung der präoperativen Simulation. 61
4.29 Funktionsprinzip der Simulationsart Punktpuls. 62
4.30 Funktionsprinzip der Simulationsart Linienpuls. 63
4.31 Pulsverteilung bei der Simulationsart Linienpuls. 63
4.32 Visualisierung der Krater- und Schnittkantenentwicklung im Geoserver. 64
4.33 Einzelschritte in der GUI zum Export der Daten und zur intraoperativen Abtragsberechnung. 65
4.34 Schematische Darstellung des Simulink-Modells zur Ansteuerung des Roboters und zum Versenden der realen Prozessparameter. 66
4.35 Einkoppeln des Roboters in eine offene und in eine geschlossene splineförmige Trajektorie 67
4.36 Interpolation zwischen zwei TCPs und daraus resultierende Wegabweichung bei der intraoperativen Abtragsberechnung. 69

5.1 Verwendete Laserobjekte im Geoserver. 71

5.2 Versuchsaufbau Genauigkeitsmessung. 73
5.3 Ergebnisse der Genauigkeitsmessung beim Modell M_Z. 74
5.4 Ergebnisse der Genauigkeitsmessung beim Modell M_G. 75
5.5 Möglichkeiten zur Beurteilung der Kraterform. 75
5.6 Kratervoxel der Modelle M_M. 76
5.7 Fehler des Modells M_Z bei schrägem Auftreffen des Laserstrahls auf eine Fläche. 77
5.8 Fehler des Modells M_Z bei schrägem Auftreffen des Laserstrahls auf den Octree. 78
5.9 Erklärung des Fehlers des Modells M_Z bei schrägem Auftreffen des Laserstrahls auf den Octree. 78
5.10 Aufbau des Experiments zur Simulationsart Linienpuls. 81
5.11 Voxelabtragsvolumen der Modelle M_M entlang einer Trajektorie. 82
5.12 Fehler des Modells M_Z bei der Simulationsart S_{LP}. 83
5.13 Aufbau des Experiments zur Evaluierung des Arbeitsablaufs. 84
5.14 Planung der verschiedenen Trajektorienarten. 84
5.15 Präoperative Abtragssimulation. 85
5.16 Durchführung der Laserosteotomie. 86

Tabellenverzeichnis

2.1 Physikalische Eigenschaften der Kompakta, entnommen aus [14], [38]. 5

4.1 Parameter des Roboters MIRO, entnommen aus [22], [31]. 26

4.2 Die DH-Parameter des MIRO. 27

4.3 Parameter des Lasers AT Fidelis, entnommen aus [17]. 28

5.1 Parameter der verwendeten Laserobjekte 71

5.2 Ergebnisse des Vergleichs zwischen den von [4], [19] ermittelten Schnitttiefen $h_{exp}(N)$ und den Schnitttiefen $\overline{h}_{W_i,M_M}(N)$ der Modelle M_M angewandt auf die drei Würfel W_i angegeben mit der Standardabweichung $\sigma_{W_i,M_M}(N)$ 74

5.3 Rechenzeit t_{calc,M_M} in Abhängigkeit von der Voxelgröße $voxsize$ der Laserobjekte 79

5.4 Rechenzeit t_{calc,M_M} in Abhängigkeit von der Gesamtvoxelanzahl N der Laserobjekte 80

5.5 Rechenzeit t_{calc,M_M} in Abhängigkeit von der Kratertiefe h 80

5.6 Zeit zur Visualisierung t_{vis,M_Z} des Kraters in Abhängigkeit von der Anzahl der zu visualisierenden Kratervoxel N 81

Eidesstattliche Erklärung

Ich erkläre hiermit eidesstattlich, dass ich die vorliegende Arbeit selbständig angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt.

Garching, den 31.03.2011

(Martin Lohmann)