Endbericht der Projektgruppe 434

GenuTEP

Lehrstuhl für Graphische Systeme
Fachbereich Informatik
Universität Dortmund

8. September 2004

Teilnehmer:

Claudia Dekomien

Martin Fischer

Petra Kersting

Stefan Kiesler

Birgit Lepper

Sebastian Meier

Carsten Rudat

Oliver Schuknecht

Florian Stößel

Björn Sunder

Lars Walczak

Thomas Wiederkehr

Betreuer:

Martin Wawro

Frank Weichert

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Pseudocodes

Mathematische Notation

Die folgende Auflistung enthält die in diesem Endbericht verwendeten mathematischen Notationen.

Symbol	Bedeutung	Anmerkung
M	Raumpunkt	dreidimensionaler Punkt, dargestellt als Großbuchstabe
m	Bildpunkt	benennt einen Punkt in der Bildebene
$\mathbf{x}$	Vektor	wird dargestellt als fettgedruckter Kleinbuchstabe; die Komponenten von $\mathbf{x}$ werden mit $x_1, ..., x_n$ bezeichnet
$\tilde{\mathbf{x}}$	homogener Vektor	Darstellung von $\mathbf{x}$ in homogenen Koordinaten
$\tilde{\mathbf{M}}$	homogener 3D-Vektor	dreidimensionale, homogene Vektoren werden durch Tilde und Fettdruck gekennzeichnet
$\mathbf{M}$	Matrix	bezeichnet durch einen fettgedruckten Großbuchstaben
$\mathbf{M}^{-T}$	$= \left(\mathbf{M}^{-1}\right)^T$	inverse, transponierte Matrix
AB bzw. Cm	Gerade	beschreibt die Gerade durch die Raumpunkte A und B bzw. den Raumpunkt C und den Bildpunkt m
s	Skalar	Ein Skalar wird wie ein Bildpunkt mit einem Kleinbuchstaben bezeichnet. Die Symbolbedeutung geht aus dem Kontext hervor.
$\overline{x}$	Mittelwert	Mittelwerte werden durch einen Querstrich kennzeichnet

Kapitel 1

Einleitung und Systemüberblick

1.1 Anforderung, Ziele und Motivation

Als Folge von Arthroseerkrankungen o.ä. kann es sein, dass das menschliche Kniegelenk komplett durch ein künstliches Kniegelenk ersetzt werden muss. Dieser, als *totale Endoprothese* (kurz: TEP) bezeichnete Gelenkersatz für das menschliche Kniegelenk, gehört zu den am häufigsten eingesetzten Prothesen in Deutschland. Aufgrund der Tatsache, dass die Prothesen einiger Patienten größere Verschleißerscheinungen aufweisen als die anderer, stellt sich die Frage, ob der Einbauwinkel der Prothese die Lebensdauer beeinträchtigt. Bei einer Knieoperation werden die Prothesenachsen oft nur näherungsweise an die Achsen des Ober- und Unterschenkelknochens ausgerichtet. Es wird vermutet, dass bei einem schlechten Einbauwinkel die Prothesen Belastungen unterliegen, die einen höheren Verschleiß der Prothese und des Knochens zur Folge haben. Wenn beim Eingriff die Prothesenachsen an den anatomischen Achsen der Knochen ausgerichtet werden, könnte die Lebensdauer der Prothesenimplantate verlängert werden.
Hier setzt die Arbeit der Projektgruppe GenuTEP an. Es wurde ein Softwaresystem entworfen und entwickelt, welches mithilfe von C-Bogen Aufnahmen die bisher nicht möglich gewesene Messung des Protheseneinbauwinkels erlaubt und zugleich die Abweichung zum optimalen Einbauwinkel berechnet. Weitere Leistugsmerkmale sind die postoperative Rekonstruktion der implantierten Prothesenachsen sowie die visuelle Gegenüberstellung derselben in Bezug zu den anatomischen Achsen des menschlichen Hüftknochens. Eine Interaktion mit dem Model wie z.B. Drehen ist ebenso möglich.

1.2 Ein kurzer Überblick über das Gesamtsystem

Im Folgenden wird kurz darauf eingegangen, welche Schritte notwendig sind um von den eingelesenen Bildern im DICOM Format zu einem rekonstruierten 3D Model und einer entsprechenden Volumenvisualisierung zu gelangen. Zu Beginn werden die von einem C-Bogen gewonnenen Bilder zur Kamerakalibierung verwendet und anschließend entzerrt. Im nächsten Schritt werden die Prothese, Knochen sowie das Hüft- und Fußgelenk segmentiert. Anschließend erfolgt die 3D Rekonstruktion und die Berechnung der Achsenabweichung.

1.3 Realisierung

Da jede Software-Entwicklung in einem vorher festgelegten organisatorischen Rahmen erfolgen sollte, musste zu Beginn der Entwicklung ein geeignetes Prozessmodell gewählt werden um eben diesen Rahmen vorzugeben. Zur Auswahl standen hier zum einen das Wasserfall- und das Prototypenmodell.

1.3.1 Entscheidung für das Prototypen-Modell

Da im Vorhinein bei vielen Teil-Problemstellungen nicht eindeutig festzulegen war, welcher mathematische bzw. logische Ansatz der effektivste ist, wurde aus Sicht der Softwareentwicklung das beste dafür geeignete Entwicklungsmodell gewählt, das so genannte Prototypenmodell. Anhand eines Prototypen lassen sich relativ früh die erzielbaren Ergebnisse deutlich machen. Dadurch war es möglich, möglichst früh ungeeignete Ansätze zu erkennen und zu verwerfen. Bei der Entwicklung des Endproduktes konnten dann grundlegende Ideen des Prototypen aufgegriffen und realisiert werden. Das Wasserfallmodell mit seinem sequentiellen Ablauf hätte an dieser Stelle zu Problemen geführt, da zu Beginn kein vollständiger Entwurf des Projektes möglich war. Die Wahl des Prototypenmodells bedeutete lediglich einen durch die Prototypen bedingten erhöhten Entwicklungsaufwand.

1.3.2 Prototypen-Modell

Bei der Software-Entwicklung mit Hilfe klassischer Prozessmodelle, wie etwa dem oben erwähnten Wasserfall-Modell, können unter Umständen Probleme auftauchen, die nicht mit diesen Prozessmodellen gelost werden können. Zum Beispiel verlangen klassische Prozessmodelle in der Definitionsphase eine vollständige Spezifizierung der Anforderungen. Der Auftraggeber ist jedoch oft nicht in der Lage seine Anforderungen an das zu entwickelnde System explizit und/oder vollständig zu formulieren. Auch existieren häufig unterschiedliche Lösungsmöglichkeiten für bestimmte Anforderungen. Diese müssten zuerst experimentell erprobt werden, bevor eine Entscheidung über die endgültige Losung getroffen werden kann. Bei beiden Varianten führt der Einsatz klassischer Prozessmodelle zu Problemen, da diese nach der Definitionsphase keine Beteiligung des Auftraggebers vorsehen bzw. der Entwicklungsablauf fest vorgegeben wurde und nicht ohne weiteres abgeändert werden kann. Diese Probleme können mit Hilfe des Prototypen-Modells gelöst werden.

Beim Prototypen-Modell wird als Vorgehensweise das so genannte „prototyping“ gewählt. Dies bedeutet, dass frühzeitig lauffähige Modelle, also Prototypen, erstellt werden. Hierbei gibt es verschiedene Arten von Prototypen, die jeweils einem anderen Zweck dienen.
Der Demonstrationsprototyp dient zur Auftragsakquisition. Er soll dem Auftraggeber vermitteln, wie das Produkt im Prinzip aussehen könnte. Nach der Erfüllung seiner Aufgabe wird dieser verworfen.
Ein Prototyp im engeren Sinne dient dazu, den Anwendungsbereich zu analysieren. Er ist ein provisorisches, ablauffähiges Software-System.
Das Labormuster demonstriert die technische Umsetzbarkeit des Produktmodells, indem es konstruktionsbezogene Fragen beantwortet. Endbenutzer nehmen im Allgemeinen nicht an der Eva-

luierung des Labormusters teil.
Ein Pilotsystem ist ein Prototyp, der selbst der Kern des Produkts ist. Ab einem gewissen Entwicklungsstand verschwindet die Unterscheidung zwischen Prototyp und Produkt immer mehr. Die Weiterentwicklung des Pilotsystems geschieht dann in Zyklen. Dabei hilft ein Pilotsystem dabei, die organisatorische Integration des Produkts vorzubereiten, indem es dem Benutzer einen Vorgeschmack auf das System gibt.

1.4 Struktur dieses Berichts

Dieses Kapitel gab einen kurzen Überblick über das Projekt. Die detaillierte Vorgehensweise wird in den Kapiteln 3 bis 6 beschrieben. Dabei stellt jedes Kapitel ein Modul des Softwaresystems vor. Im Einzelnen handelt es sich dabei um Datenakquisition, Kalibrierung und Dewarping, Segmentierung, 3D-Rekonstruktion sowie Persistenz. Die bei der Realisierung verwendeten Werkzeuge – wie zum Beispiel Qt oder Newmat – werden in Kapitel 7 vorgestellt. Das abschließende Fazit befindet sich in Kapitel 8, das Benutzerhandbuch in Kapitel 9. Der Anhang besteht aus den Klassendiagrammen, dem Pflichtenheft und dem Qualitätsmanagement-Handbuch.

Kapitel 2

Datenakquisition

Die zur Rekonstruktion benötigten Bilddaten können in zwei Formaten vorliegen, einmal im DICOM-Format, welches außer den reinen Bilddaten auch zusätzliche Informationen über den Patienten, den zur Aufnahme verwendeten C-Bogen, etc. enthält und außerdem im PNG-Format, welches neben den Bilddaten keine zusätzlichen Informationen enthält.

2.1 Datenerfassung

2.1.1 C-Bogen

Der C-Bogen wurde in den 70er Jahren entwickelt und ist kostengünstiger und platzsparender als der Computer Tomograph. Jedoch ist die Bildqualität von C-Bögen schlechter und dieser kann zur Zeit nicht zu 3D-Rekonstruktion eingesetzt werden.
Eingesetzt wird der C-Bogen oft für intraoperative Durchleuchtung. Der C-Bogen hilft während einer Untersuchung oder bei einer Operation Veränderungen an Organen zu erkennen. Darüber hinaus dient er der Lagekontrolle von Implantaten und technischen Gerätschaften.

Bei dem, wie der Name schon sagt, c-förmigen Gerät (siehe Abbildung 2.2) wird das zu durchleuchtende Gewebe zwischen Röntgenröhre und Bildverstärker platziert. Angeschlossen an den Bildverstärker ist eine Kamera, meistens eine CCD-Kamera (Charge-Coupled Device), die das Licht in ein elektrisches Signal umwandelt und anschließend als Videosignal verschlüsselt. Anschließend wird das Bild zu einem Bildschirm weitergeleitet und kann dort als Echtzeitbild betrachtet werden. Der C-Bogen wird (bei den konventionellen Geräten) manuell bewegt und kann um ca. 150° um das Gewebe herum rotieren. Der Vorteil gegenüber den CT-Bildern ist die niedrigere Strahlenbelastung sowie die bessere Mobilität[G$^+$03]. Nachteile des C-Bogens sind unter anderem:

- Notwendigkeit von Kalibrierung
- verzerrte Bilder
- Verlust von Tiefe und Volumen
- limitiertes Sichtfeld
- limitierte Rotation

Aufgrund dieser Nachteile sind C-Bögen keine vollwertige Alternative zu den Computer Tomographen [Fic01].

Der C-Bogen bedient sich der Technik der Fluoroskopie, die es ermöglicht Röntgenbilder in Echtzeit während einer Untersuchung zu erhalten. Der C-Bogen erfüllt alle Anforderungen, die die Fluoroskopie an eine medizinische technische Modalität stellt.

Das konventionelle Fluoroskopie-System besteht aus:

- einer Röntgenröhre
- einem Untersuchungstisch
- einer fluoreszierenden Platte gekoppelt mit einem Bildverstärker
- einem Video-System

Weil eine Untersuchung mit der Fluoroskopiemethode wesentlich länger dauert als eine normale Röntgenuntersuchung, sollte die Belichtungszeit für die einzelnen Fluoroskopieaufnahmen geringer sein. Diese geringere Belichtungszeit hat zur Folge, dass das Fluoroskopiebild mit wesentlich weniger Röntgenphotonen gebildet wird. Dies wirkt sich nachteilig auf die Bildqualität aus. Zur Verstärkung von Kontrasten werden häufig Kontrastmittel wie Barium (z.B. im Magen-Darm-Trakt) eingesetzt [Ima03, PMM03].

Bei der digitalen Fluoroskopie wird das entstehende analoge Videosignal digitalisiert, im Rechner gespeichert, bearbeitet und anschließend auf dem Bildschirm angezeigt. Darüber hinaus können die Daten der Untersuchung bei Bedarf über ein Netzwerk verschickt werden [PMM03].
Folgende Schritte werden bei der Digitalisierung durchlaufen:

1. Umwandlung des Röntgenlichtes in ein Videosignal unter Berücksichtigung der Kamera-Kalibrierung
2. Kompression und Digitalisierung der Daten

Um das, vom C-Bogen aufgenommende Bild im Rechner zu speichern und anschließend zu bearbeiten, muss das von der Kamera gelieferte analoge Signal in digitale Daten umgewandelt werden. Hierfür setzt man FrameGrabber ein, die als Plugin-Karten für PC's und Laptops erhältlich sind. Das ermöglicht eine Datenübertragung in Echtzeit. Der FrameGrabber empfängt also das analoge Signal der CCD-Kamera und wandelt dieses in digitale Daten um. Diese werden dann über den PCI-Bus direkt zum Hauptspeicher transportiert. Mittlerweile gibt es C-Bögen, die ihre Bilddatensätze nicht in analoger, sondern direkt in digitaler Form ausgeben und damit einen zusätzlichen Framegrabber überflüssig machen.

Spätestens mit dem Aufkommen der Idee einer digitalen Archivierung von Bildern (PACS) und einer elektronischen Bildverteilung im Krankenhaus entstand das Bedürfnis, digitale Bilder zwischen Geräten verschiedener Hersteller austauschen zu können. Ein Standardformat wurde von der 'National Electrical Manufacturers Association' (NEMA) und dem 'American College of Radiology' (ACR) entwickelt. So entstand 1993 das DICOM ("Digital Imaging and Communications in Medicine") -Standard-Format (siehe Abschnitt 2.2.1), eine offene (herstellerunabhängige) Kommunikationsplattform für medizinische Bilder und bildbezogene Informationen [Uni03e].

Der Inhalt des DICOM-Standards geht weit über die Definition eines reinen Austauschformats für medizinische Bilddaten hinaus.

2.1.2 Kalibriergegenstände

Das in dieser Projektgruppe entwickelte Programm greift auf Daten von drei Kalibriergegenständen (Kalibriergitter, Kalibrierstab, Kalibrierquader) zurück. Diese Daten werden in verschiedenen Phasen des Programmes ausgewertet.

Das Kalibriergitter (siehe Abbildung 2.1) besteht aus einer dursichtigen Kunststoffplatte, in der mehrere Wolframdrähte eingearbeitet sind, die gitterförmig angeordnet sind. Dieses Gitter muss im ersten Schritt vor den Detektor des C-Bogens montiert und aufgenommen werden. Danach wird das Gitter auf den Tisch gelegt, um mehrere Aufnahmen aus verschidenen Blickwinkeln zu akquirieren. Diese Aufnahmen werden in der Kalibrierungsphase für die Videokamera des C-Bogens dafür benötigt, um die intrinsische Matrix und eine Matrix von Verschiebevektoren der Bildverzerrung zu gewinnen (siehe Kapitel 3). Die Verschiebevektoren werden verwendet, um die Bilder infolge der sphärischen Aberration durch die Linse der Videokamera zu entzerren. Die intrinsische Matrix, die Informationen über das Auflösungsvermögen und über das Bildseitenverhältnis der Videokamera beinhaltet, bildet eine Voraussetzung für eine euklidische dreidimensionale Rekonstruktion (siehe Kapitel 5).

Für eine solche Rekonstruktion müssen Fluoroskopieaufnahmen für die drei anatomischen Bereiche Hüfte, Knie und Knöchel angefertigt werden. In Abbildung 2.2 wird der Versuchsaufbau mit einem künstlichen Kniegelenk, an dem eine Prothese angebracht ist, dargestellt. Dabei müssen der Stab und der Quader sehr dicht neben das Bein gelegt werden, damit sie vom C-Bogen möglichst vollständig abgebildet werden, da der Sichtbereich bei vielen C-Bögen sehr eingeschränkt ist. Der Stab muss darüber hinaus während der gesamten Aufnahmeprozedur (für alle drei Aufnahmesequenzen für Hüfte, Knie, Knöchel) in derselben Position liegen bleiben. Der Quader wird für eine Aufnahmesequenz neben entsprechenden anatomischen Bereich gelegt und darf während dieser Aufnahmesequenz nicht verschoben werden bis zum nächsten anatomischen Abschnitt übergegangen werden kann (siehe Abbildung 2.3 und 2.4). Der Aufbau und Zweck des Quaders und des Stabes wird im Folgenden beschrieben.

Der Kalibrierquader besteht aus einem quaderförmigen Block Acryl, in dem zwölf Metallkugeln je mit einem Durchmesser von zwei Millimetern eingebettet sind (siehe Abbildung 2.5). Für die Kugeln wurden in den Quader Löcher gebohrt, die (nach der Einbettung der Kugeln) mit Epoxidharz ausgegossen worden sind. Aufgrund der diamantförmigen Metallkugelanordnung wird im Folgenden oft das Synonym „Kalibrierungsdiamant“ für den Kalibrierquader verwendet. Mit diesem Kalibrierquader ist es möglich eine euklidische Rekonstruktion der Kamerapositionen und der geometrischen Form (Anordnung der zwölf Metallkugeln) zu berechnen. Der Algorithmus erfordert für jeden Bereich mindestestens drei Aufnahmen, in denen alle zwölf Kugeln abgebildet sind. Dann ist eine Rekonstruktion überhaupt möglich (siehe Kapitel 5). Außerdem muss der Segmentierungsschritt die Kugeln in einer zwischen den Aufnahmen korrespondierenden Reihenfolge erkennen. Jeder der zwölf Punkte des Kalibrierquaders wurde eindeutig indiziert. Dadurch ist es möglich einem Punkt, dem in einem Bild der Index i zugeordnet worden ist, denselben Index i in einem anderen Bild zuzuordnen (siehe Kapitel 4). Dies und die Kenntnis über die originale Form (samt Abmessungen) bilden das Domänenwissen, um eine eindeutige Lösung

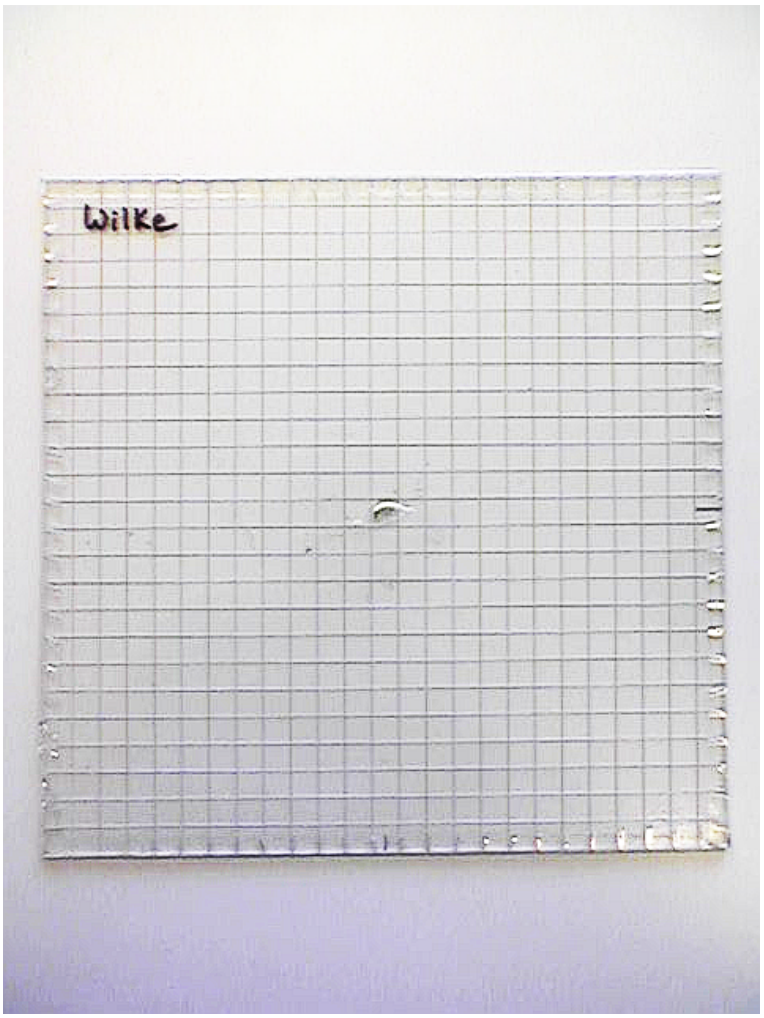


Abbildung 2.1: Das Kalibriergitter wird für die Berechnung von Verschiebevektoren zur Kompensierung der sphärischen Aberration benötigt.

Abbildung 2.2: Der Kalibrierquader und der Stab werden zusammen mit dem künstlichen Knochen (an dem die Prothese angebracht ist) von dem C-Bogen aufgenommen.

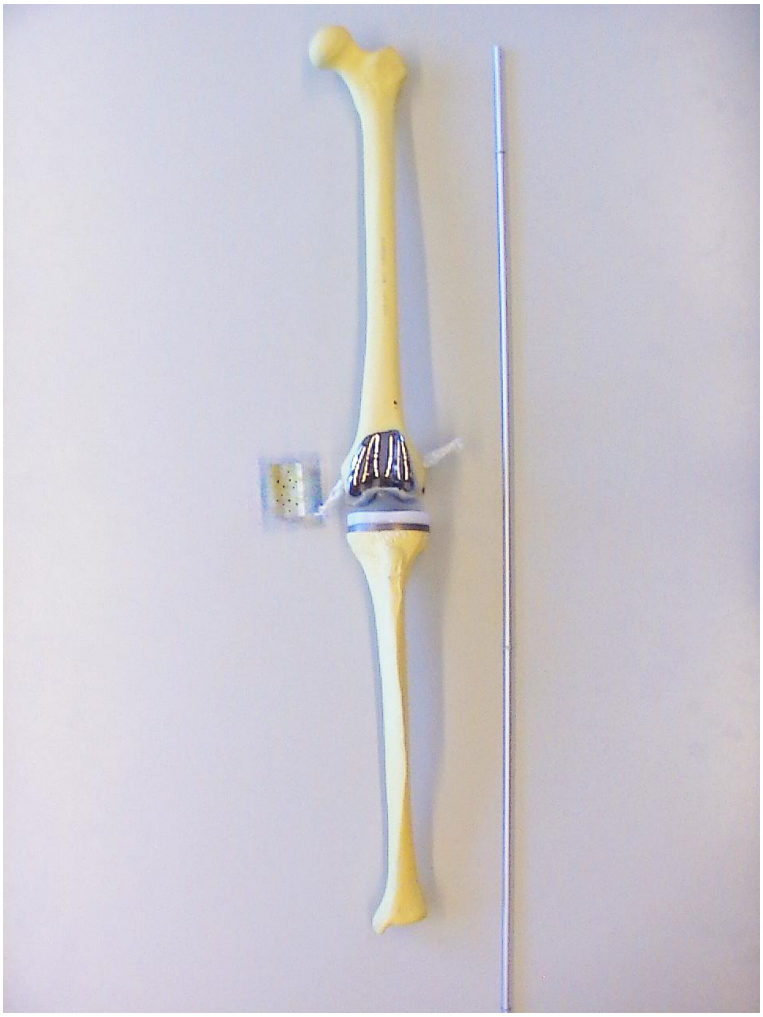


Abbildung 2.3: Der Stab wird neben den Knochen gelegt und darf während der gesamten Aufnahmeprozedur nicht verschoben werden. Der Quader/Diamant wird für jeden anatomischen Bereich, der aufgenommen wird, entweder neben die Hüfte, neben das Knie oder neben den Knöchel gelegt. Für die Abbildung wurde der Quader neben das Knie gelegt.

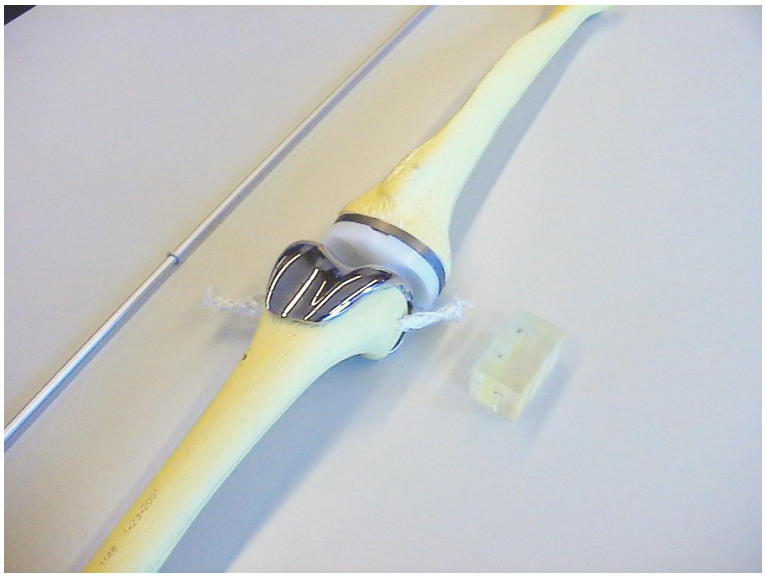


Abbildung 2.4: Der Kalibrierquader und der Stab müssen sehr dicht neben den Knochen gelegt werden, damit sie während der Aufnahmeprozedur möglichst vollständig in der Fluoroskopieaufnahme abgebildet werden.

für die Rekonstruktion zu berechnen.

Der Kalibrierstab ist eine schmale Metallröhre, an der drei Metallringe als Markierungen angebracht sind und ist bei dem Registrierungsschritt behilflich, in dem die dreidimensional rekonstruierten Bereiche ineinander transformiert werden (siehe Abschnitt 5.7). Dafür müssen im Segmentierungsschritt in mindestens drei Bildern pro antaomischen Abschnitt die Markierungen erkannt worden sein (siehe Kapitel 4), sodass eine Rekonstruktion überhaupt möglich ist. Der Kalibrierstab muss während der ganzen Aufnahmeprozedur in derselben Position liegen bleiben.

2.2 Datenformate

2.2.1 DICOM

Das „Digital Imaging and Communications in Medicine Format“, kurz DICOM, das von der National Electrical Manufacturers Association (NEMA) entwickelt wurde, um medizinische Aufnahmen zu vereinfachen und besser verteilen zu können.
Jede DICOM-Datei beginnt mit einem 128 Byte langem Header, gefolgt von den 4 Bytes ’D’, ’I’, ’C’, ’M’ und den sogenannten Tags. Ein Tag besteht aus zwei 16-bit Werten, welche den Inhalt des Tags angeben, einem 32-bit Wert der die Länge des Tag-Inhaltes angibt und dem eigentlichen Taginhalt. In den einzelnen Tags sind Informationen über die Bildgröße, das verwendete Aufnahmegerät, Patientenname, verwendetes Bildformat, etc. enthalten. Einer der Tags enthält die eigentlichen Bilddaten. Wie schon erwähnt können die Bilddaten in verschiedenen Formaten gespeichert sein. So kann z.B. verlustbehaftetes JPEG verwendet werden, aber die Bilddaten können auch verlustfrei abgespeichert werden, z.B. als Rohdaten.
GenuTEP unterstützt zur Zeit nur ein proprietäres DICOM-Format der Firma Siemens. Bei die-

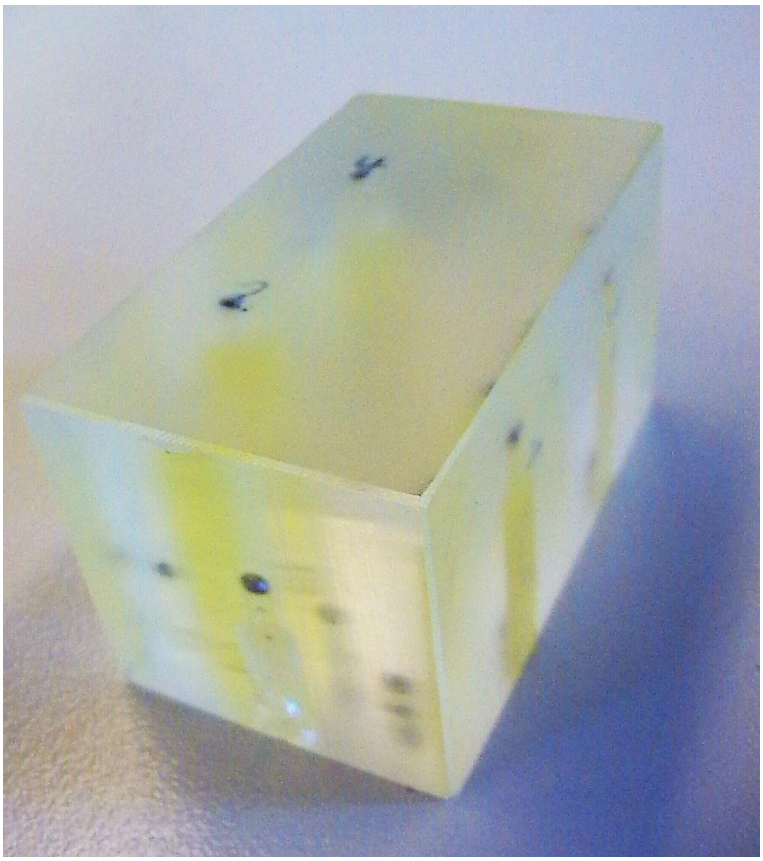


Abbildung 2.5: Der Kalibrierquader besteht aus einem Block Acryl mit zwölf eingebetteten Metallkugeln.

Abbildung 2.6: Der Kalibrierstab ist eine schmale Metallröhre, an der drei Metallringe als Markierungen angebracht sind.

sem Format fehlt der DICOM-Header komplett, die Dateien bestehen ausschließlich aus den Tags und den Rohdaten des Bildes.

2.2.2 PNG

Das Portable Network Graphics Format, kurz PNG, stellt eine Alternative zum GIF-Format dar. Im Gegensatz zum GIF-Format kann es frei benutzt werden ohne Abgaben leisten zu müssen. Es erlaubt das verlustfreie Komprimieren und Speichern von Bildern der unterschiedlichsten Formate, wie etwa Graustufen- und Truecolor-Bildern. Von GenuTEP unterstützt werden zur Zeit die Graustufen-Formate in 8- bzw. 16-bit Farbtiefe. Zum Einlesen und auch Speichern der Bilder wird eine frei verfügbare PNG-Bibliothek, die LibPNG, verwendet. Diese übernimmt das Lesen, Schreiben, Komprimieren und Dekomprimieren der Bildinformationen, sowie das Überprüfen der Datei auf gültige Daten.
In den PNG-Dateien befinden sich im Gegensatz zu den DICOM-Dateien keine zusätzlichen Informationen, wie etwa Patientenname, Aufnahmezeit, etc. So müssen die Patientendaten von Hand eingegeben werden und die Sortierung der Bilder ist nicht automatisch nach der Aufnahmezeit möglich. Die PNG-Dateien werden aus einem durch eine Framegrabberkarte aufgezeichnetem Film gewonnen und haben in diesem Fall 8-bit Farbtiefe. Die Grabbing-Funktionalität ist jedoch nicht in GenuTEP integriert sondern muss von einem externen Programm übernommen werden.
16-bit PNG-Dateien liegen lediglich dann vor, falls ein bestehendes Projekt geöffnet wird und die ursprünglichen Bilder im DICOM-Format mit 12- oder 16-bit Farbtiefe vorlagen, da diese nach dem Dewarping als PNG-Dateien gespeichert werden.

2.2.3 Einlesen der Bilddaten

Die PNG- bzw. DICOM-Dateien müssen in einem Ordner in drei Unterordnern, mit den Namen „Huefte“, „Knie“ und „Knoechel“, aufgeteilt vorliegen. Der Benutzer hat für diese Vorsortierung zu sorgen, da lediglich aus DICOM-Dateien die nötigen Informationen gewonnen werden könnten um eine automatische Verteilung der Bilder auf die drei Ordner zu gewährleisten. Falls in einem Ordner DICOM- und PNG-Dateien gleichzeitig vorkommen sollten werden lediglich die DICOM-Dateien eingelesen, da bei diesen erstens von einer höheren Bildqualität gegenüber PNG-Dateien ausgegangen werden muss und zweitens nützliche Zusatzinformationen enthalten sind, die z.B. eine automatische Sortierung der Bilder nach der Aufnahmezeit ermöglichen.
Bei den unterstützten DICOM-Dateien handelt es sich wie bereits oben erwähnt um ein proprietäres Format der Firma Siemens. Zum verifizieren ob es sich bei diesen Dateien um das unterstützte DICOM-Format handelt werden alle Tags ausgelesen. Für GenuTEP wichtige Tags, wie etwa der Patientenname, werden ausgewertet und gespeichert. Falls kein Tag für die Bilddaten gefunden wird, handelt es sich um keine gültige DICOM-Datei und die Datei wird verworfen. Falls keine einzige gültige DICOM-Datei gefunden wurde werden alle Dateien erneut überprüft, diesmal werden allerdings PNG-Dateien gesucht.
Hierzu wird der Header ausgelesen. Falls dieser gültig ist wird zusätzlich überprüft ob es sich um Graustufen-PNG-Dateien handelt, ansonsten wird die Datei erneut verworfen.
Wurden gültige DICOM- oder PNG-Dateien gefunden werden diese eingelesen und in einer neu definierten Datenstruktur zwischengespeichert. Diese Datenstruktur ist das GenuImage (vgl. Kapitel A.3), welches nur die Bilddaten enthält. Nach dem Dewarping der GenuImages werden diese als PNG-Dateien auf Festplatte gespeichert. Die GenuImages werden nun in einer Projekt-

datenstruktur gespeichert. Diese behält jedoch lediglich eine fixe Anzahl an Bildern gleichzeitig im Speicher, um sicherzustellen das dem Programm immer genügend freier Arbeitsspeicher zur Verfügung steht. Nicht im Speicher vorhandene GenuImages werden je nach Bedarf automatisch aus den dewarpten PNG-Dateien nachgeladen und nicht benötigte GenuImages automatisch aus dem Speicher entfernt.

Kapitel 3

Kalibrierung / Dewarping

3.1 Kalibrierung

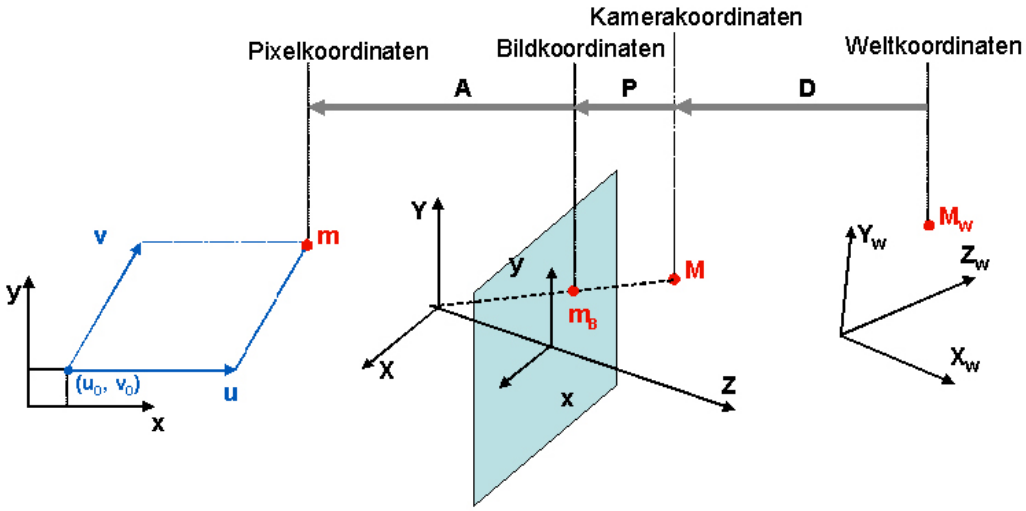


Abbildung 3.1: Definition des Bildkoordiantensystems

Für die 3D-Rekonstruktion sind die intrinsischen Parameter der Kamera (Brennpunkt und Skalierungsfaktoren in u- und v-Richtung. Die Definition von u und v ist in Abbildung 3.1 zu sehen) nötig. Die Berechnung dieser Parameter erfolgt durch die Kalibrierung der Kamera. Hierfür wird ein in [Zha98] vorgestelltes Verfahren benutzt. Dieses Verfahren benutzt ein planares Kalibrierungsobjekt dessen genaue Maße bekannt sein müssen. Das Verfahren hat den Vorteil, dass die genaue Lage des Kalibrierungsobjekts und die Lage der Kamera im Raum nicht bekannt sein müssen. Zur Berechnung der intrinsischen Parameter sind mindestens drei Aufnahmen des Kalibrierungsobjekts, nötig, da sonst das Gleichungssystem (3.18) unterbestimmt ist. Die Aufnahmen müssen aus verschiedenen Winkeln (zwischen 60°und 120°) erfolgen. Erfolgt keine Drehung der Kamera, können aus den Gleichungen (3.7) und (3.8) keine neuen Bedingungen abgeleitet werden [Zha98]. Die maximale Anzahl der Bilder wurde im Projekt auf 4 beschränkt. Aus den Bildern werden die Koordinaten der Gitterpunkte extrahiert (Kapitel 3.2.2) und unter Verwendung eines Modells des Kalibrierungsgitters kann die gesuchte Matrix berechnet werden.

3.2 Kalibrierung

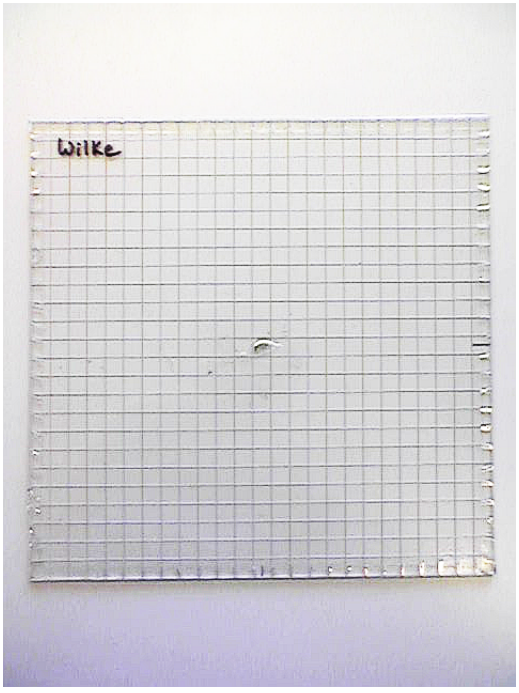


Abbildung 3.2: Kalibrierungs-Gitter

3.2.1 Das Kalibrierungsobjekt

Zur Kalibrierung wird ein rechtwinkliges äquidistantes Gitter (siehe Abbildung 3.2) verwendet. Es wurde in der Mechanischen Werkstatt des Universitätsklinikums Essen für die Projektgruppe 434 (3D-IVUS View) gefertigt. Im Abstand von 1cm wurden Nuten in eine Plexiglasscheibe gefräst und darin Wolfram der Stärke 0,36mm eingelegt. In der Mitte des Gitters ist ein L-förmiger Marker aus Kupferdraht angebracht, der eine Orientierung in den Aufnahmen ermöglicht.

3.2.2 Gitterpunke erfassen

Um die Berechnung durchführen zu können, müssen die Koordinaten der Gitterpunkte in jedem Bild erfasst werden. Hierfür wurde eine automatische Gitterpunktsuche implementiert. Dabei wurden unterschiedliche Verfahren zur Kantendetektion miteinander kombiniert. Folgende Schritte werden durchlaufen:

1. Kanten im Bild finden
2. Zwischenräume füllen
3. Skelettierung anwenden
4. Gitterpunkte bestimmen

Kanten finden Zur Kantenextraktion wurde auf den Sobel-Operator zurückgegriffen. Diese Operation liefert ein Gradientenbild, in dem nur die Übergänge von hell auf dunkel oder von dunkel zu hell als Kanten dargestellt sind.

Zwischenräume füllen Der Sobeloperator liefert eine Kante für jeden Übergang von hell auf dunkel und dunkel auf hell. Dabei entsteht ein Zwischenraum zwischen den Kanten. Dieser Zwischenraum wird mittels Dilatation gefüllt. Als strukturgebendes Element wurde ein Viereck der Größe 3x3 verwendet.

Skelettierung anwenden An dieser Stelle kommt der Skelettierungsalgorithmus von Zhang und Suen [GW92] zum Einsatz. Ziel dieses Schrittes ist es, ein Gitter zu erhalten, in dem alle Gitterlinien genau einen Pixel breit ist.

Gitterpunkte bestimmen Sei p ein Pixel in einem Bild. Jeder Punkt mit $s(p) > 2$ liegt an einer Kreuzung und wird als potentieller Gitterpunkt erfasst. Aus eng zusammenliegenden Gitterpunkten wird anschließend der Durchschnitt gebildet. Die so gefundenen Gitterpunkte werden dann im Programm übernommen und können vom Benutzer korrigiert werden.

3.2.3 Extraktion der intrinsischen Parameter

3.2.3.1 Grundlagen

Es wird ein Lochkameramodell angenommen. Die Projektion eines Punktes im Raum auf die Bildebene lässt sich dabei wie folgt beschreiben:

$$\mathbf{s} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = [\mathbf{R}\ \mathbf{t}] \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{3.1}$$

Dabei bezeichnet s einen nicht näher bestimmten Skalar. Die intrinsische Parametermatrix **A** ist gegeben durch:

$$\mathbf{A} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

Der Brennpunkt ist hierbei gegeben durch (u_0, v_0), α und β stellen die Skalierungsfaktoren in u- und v-Richtung dar. Der Parameter γ gibt die Verschiebung der Bildachsen an. Die extrinsischen Parameter werden durch die Rotationsmatrix und die Translation

$$\mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \text{ und } \mathbf{t} = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \tag{3.3}$$

beschrieben.
Unter der Annahme, dass im Weltkoordinatensystem für alle Punkte des Modells $z = 0$ gilt,

lässt sich Gleichung (3.1) wie folgt vereinfachen:

$$\mathbf{s}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{A}\,[\mathbf{r}_1 \;\; \mathbf{r}_2 \;\; \mathbf{t}] \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{3.4}$$

mit $\mathbf{r}_i$ für die i-te Spalte von **R**. Somit ergibt sich für die Abbildung **H** vom Modell auf die Bildebene

$$\mathbf{s}m = \mathbf{HM} \;\text{ mit } \mathbf{H} = [\mathbf{h}_1 \;\; \mathbf{h}_2 \;\; \mathbf{h}_3] \;=\; \mathbf{A}[\mathbf{r}_1 \;\; \mathbf{r}_2 \;\; \mathbf{r}_3] \tag{3.5}$$

Aus Gleichung (3.5) ergibt sich

$$[\mathbf{h}_1 \;\; \mathbf{h}_2 \;\; \mathbf{h}_3] \;=\; \lambda A[\mathbf{r}_1 \;\; \mathbf{r}_2 \;\; \mathbf{t}] \tag{3.6}$$

mit einem willkürlichen skalaren Faktor λ, da die Projektion 3.1 nur bis auf einen skalaren Faktor s eindeutig ist.
Aus der Tatsache, dass $\mathbf{r}_1$ und $\mathbf{r}_2$ orthogonal sind, ergeben sich für jedes Bild folgende 2 Gleichungen

$$\mathbf{h}_1^T(\mathbf{A}^{-1})^T\mathbf{h}_2 \;=\; 0 \tag{3.7}$$
$$\mathbf{h}_1^T(\mathbf{A}^{-1})^T\mathbf{h}_1 \;=\; \mathbf{h}_2^T(\mathbf{A}^{-1})^T\mathbf{h}_2 \tag{3.8}$$

Da die Abbildung acht Freiheitsgrade besitzt und es sechs extrinsische Parameter gibt (drei aus Rotation und drei aus Translation), ergeben sich zwei Einschränkungen für die intrinsischen Parameter [Zha98].

3.2.3.2 Finden der Abbildung von Modell auf Bild

Die Abbildung **H**, die Gleichung (3.5) erfüllen sollte, kann durch Minimierung von

$$\sum_i (m_i - \widehat{m}_i)^T \wedge_{m_i}^{-1} (m_i - \widehat{m}_i) \tag{3.9}$$

mit

$$\widehat{m}_i = \frac{1}{\overline{h}_3 M_i}\begin{pmatrix} \overline{h}_1 M_i \\ \overline{h}_3 M_i \end{pmatrix} \text{ mit } \overline{h}_i \text{ für die i-te Zeile von } \mathbf{H} \tag{3.10}$$

gefunden werden [Zha98]. Hierbei handelt es sich um ein nichtlineares Minimierungsproblem, das mit der Levenberg-Marquardt-Methode gelöst werden kann. Die benötigte initiale Schätzung ergibt sich aus den Gleichungen (3.11) und (3.12). Sei $x = [h_1, h_2, h_3]^T$. Die Gleichung (3.5) lässt sich dann umschreiben in

$$\begin{pmatrix} M^T & 0^T & -uM^T \\ 0^T & M^T & -vM^T \end{pmatrix} x = 0. \tag{3.11}$$

Sind n Punkte gegeben, so ergeben sich n derartige Gleichungen, welche sich in der Form

$$\mathbf{L}x = 0 \tag{3.12}$$

zusammenfassen lassen. Die Matrix **L** hat dabei die Dimension $2n \times 9$. Die Lösung von 3.12 ist gegeben durch den Eigenvektor des kleinsten Eigenwertes von $\mathbf{L}^T\mathbf{L}$ [Zha98].

3.2.3.3 Schätzung der intrinsischen und extrinsischen Parameter

Sei

$$\mathbf{B} = \mathbf{A}^{-T}\mathbf{A}^{-1} = \begin{pmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0^2}{\beta^2} & -\frac{(v_0\gamma - u_0\beta)^2}{\alpha^2\beta^2} - \frac{v_0^2}{\beta^2} + 1 \end{pmatrix} \tag{3.13}$$

Offensichtlich ist **B** symmetrisch und somit durch einen Vektor

$$\mathbf{b} = [b_{11}, b_{12}, b_{22}, b_{13}, b_{23}, b_{33}]^T \tag{3.14}$$

eindeutig definiert. Es gilt

$$\mathbf{h}_i\mathbf{B}\mathbf{h}_j = v_{ij}^T\mathbf{b} \tag{3.15}$$

mit

$$v_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}, h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}, h_{i3}, h_{i3}h_{j3}]^T \tag{3.16}$$

Somit lassen sich Gleichung (3.7) und (3.8) bei einer gegebenen Abbildung als zwei Gleichungen in **b** darstellen

$$\begin{pmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{pmatrix} \mathbf{b} = 0 \tag{3.17}$$

Sind n Bilder gegeben, ergibt sich aus n Gleichungen wie in (3.17) ein Gleichungssystem

$$\mathbf{V}\mathbf{b} = 0 \tag{3.18}$$

mit einer $2n \times 6$ Matrix **V**, welche für $n \geq 3$ eine eindeutige Lösung **b** im *Least-Squares*-Sinne besitzt (bis auf einen skalaren Faktor λ). Die Lösung von Gleichung (3.18) lässt sich darstellen als der Eigenvektor zum kleinsten Eigenwert von $\mathbf{V}^T\mathbf{V}$. Aus **B** lassen sich die Koeffizienten von **A** direkt berechnen.

(3.19)

$$v_0 = (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2) \tag{3.20}$$
$$\lambda = B_{33} - [B_{12}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \tag{3.21}$$
$$\alpha = \sqrt{\lambda/B_{11}} \tag{3.22}$$
$$\beta = \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)} \tag{3.23}$$
$$\gamma = -B_{12}\alpha^2\beta/\lambda \tag{3.24}$$
$$u_0 = \gamma v_0/\beta - B_{13}\alpha/\lambda \tag{3.25}$$

Ist **A** bekannt, lassen sich die extrinsischen Parameter für jedes Bild berechnen.

$$\lambda_1 = 1/\left\|\mathbf{A}^{-1}h_1\right\| = 1/\left\|\mathbf{A}^{-1}h_1\right\| \tag{3.26}$$
$$r_1 = \lambda_1\mathbf{A}^{-1}h_1 \tag{3.27}$$
$$r_2 = \lambda_1\mathbf{A}^{-1}h_2 \tag{3.28}$$
$$r_3 = r_1 \times r_2 \tag{3.29}$$
$$t = \lambda_1\mathbf{A}^{-1}h_3 \tag{3.30}$$

3.2.3.4 Verbessern der Parameter

Ausgehend von n Bildern mit m Punkten können die oben gefundenen Parameter durch Minimierung des folgenden Funktionals verbessert werden

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \|m_{ij} - \widehat{m}(\mathbf{A}, \mathbf{R}_i, t_i, M_j)\|^2 . \tag{3.31}$$

Hierbei stellt m_{ij} den j-ten Punkt im i-ten Bild dar, $\widehat{m}(\mathbf{A}, \mathbf{R}_i, t_i, M_j)$ ist die Projektion von M_j im Bild i gemäß Gleichung (3.4). Auch dieses nichtlineare Minimierungsproblem kann mit der Levenberg- Marquardt-Methode gelöst werden. Die initiale Schätzung ergibt sich aus den obigen Lösungen.

3.3 Dewarping

Die vom C-Bogen aufgenommenen Röntgenbilder weisen den so genanten “Barrel“ Effekt auf. Das bedeutet, dass die Bilder vor allem an den Seiten verzerrt sind. Gründe hierfür sind Abweichungen in der Linse der C-Bogen-Kamera und Beinflussung durch das Magnetfeld der Erde. Diese Verzerrungen des Bildes führen zu erheblichen Abweichungen bei der Bestimmung der Projektionsmatrizen und Kameraparameter im Laufe des Algorithmus. Daher müssen die Bilder mit entsprechenden Dewarping Algorithmen [Rup92] entzerrt werden. Hierzu werden spezielle Aufnahmen von einem speziellen Gitter (s.Kapitel 3.2.1) benötigt, die mit dem entsprechenden C-Bogen gemacht werden müssen. Das Gitter wird vor der Kamera des C-Bogens angebracht, dann wird eine normale Röntgenaufnahme angefertigt. Mit den Informationen, die man aus diesem Bild mit Hilfe des Dewarping Algorithmus erhält können alle Bilder, die mit diesem C-Bogen aufgenommen wurden, entzert werden.

3.3.1 Dewarping Materialien

Zum Entzerren benötigt man genauso wie für die Kalibration ein rechtwinkliges Gitter (siehe Kapitel 3.2.1 auf Seite 34).

3.3.2 Der Dewarping-Algorithmus

Das Prinzip des Dewarping Algorithmus ist folgendermaßen:

- Bestimmen der verzerrten Punkte des Gitters, siehe Kapitel 3.2.2 auf Seite 34
- Bestimmung der korrespondierenden Referenzgitterpunkte
- Berechnen der Verschiebevektoren der Gitterpunkte zu den Punkten des Referenzgitters
- Für jeden Pixel wird entsprechend der Verschiebevektoren der Gitterpunkte ein Verschiebevektor durch Interpolation berechnet.
- Die einzelnen Pixel werden nun gemäß der vorher berechneten Verschiebevektoren verschoben und es wird so ein neues, unverzerrtes Bild aufgebaut.

3.3.3 Bestimmung der korrespondierenden Referenzgitterpunkte

Bei den Bildern kann man davon ausgehen, dass das Bild in der Mitte nur gering verzerrt ist. Daher ist es möglich, aus den Gitterabständen der mittigsten Punkte des Gitters das restliche Referenzgitter zu rekonstruieren. Hierzu wird zunächst der Mittelpunkt des Bildes berechnet. Anschließend werden die vier Gitterpunkte, die den geringsten Abstand zu dem Mittelpunkt haben, bestimmt. Dann werden die Abstände d dieser vier Punkte zueinander berechnet. Es wird dann ein Referenzgitter aufgebaut, dessen Gitterabstände genau d entspricht. So wird für jeden der n Punkte P_i des verzerten Gitters ein Referenzpunkt R_i (Stützstelle) berechnet.

3.3.3.1 Berechnen des Verschiebevektoren der Gitterpunkte

Hierzu wird lediglich eine einfache Abstandsberechnung der Punkte zu ihren zugewiesenen Referenzpunkten durchgeführt. Mit dem Gitterpunkt $\mathbf{P}_i(p_i(x), p_i(y))$ und dem Referenzgitterpunkt $\mathbf{R}_i(r_i(x), r_i(y))$ wird der Verschiebevektor $\mathbf{v}_i(v_i(x), v_i(y))$ mit i = 1,...,n folgendermaßen berechnet.

$$v_i(x) = (p_i(x) - r_i(x)) \tag{3.32}$$

$$v_i(y) = (p_i(y) - r_i(y)) \tag{3.33}$$

3.3.3.2 Radiale Basisfunktion

Zum Berechnen der Verschiebevektoren der einzelnen Pixel werden radiale Basisfunktionen verwendet. Hierbei wird die Interpolationsfunktion aus einer Linearkombination von Basisfunktionen konstruiert, um anschließend die Koeffizienten α_i der Basisfunktionen

$$f(x) = \sum_{i=1}^{n} \alpha_i \mathbf{R}(d_i(x)) \tag{3.34}$$

zu ermitteln.
Die Werte der Funktionen in **R** hängen nur von den Abständen der Punkte (Pixel) zu den Stützstellen des Gitters ab, daher sind die einzelnen Basisfunktionen $\mathbf{R}_i$ radialsymmetrisch. Die Koeffizienten α_i werden durch Einsetzen der Stützstellen und das darauf folgende Lösen des so entstehenden Gleichungssystems, berechnet.
Sehr bekannte radiale Basisfunktion sind die der Hardyschen Multiquadriken:

$$R(d_i(x)) = (d_i^2 + r_i^2)^\mu \text{ mit } r > 0 \text{ und } \mu \neq 0. \tag{3.35}$$

Um das Lösen des Gleichungssystems stets zu gewährleisten wurde der Exponentμ auf $\mu = -1$ gesetzt. Um gute Ergebnisse zu erhalten, muss r kritisch gewählt werden. Aus verschiedenen Quellen [Rup92] [Rup95] wurde ersichtlich, dass das Berechnen von individuellen r_i für jeden Datenpunkt P_i notwendig ist. Hierbei ist r_i der kürzeste Abstand von P_i zu seinem nächsten Nachbarn:

$$r_i = \min_{i \neq j} d_i(x_j) \tag{3.36}$$

Kapitel 4

Segmentierung

4.1 Was ist Segmentierung?

Unter „Segmentierung“ versteht man in der digitalen Bildverarbeitung Verfahren, die ein komplexes Gesamtbild in mehrere für die weitere Verarbeitung interessante Teilbereiche unterteilen. Man unterscheidet grundsätzlich zwei Arten von Segmentierung: Die vollständige und die unvollständige Segmentierung. Die vollständige Segmentierung lässt sich sehr gut durch folgende formale Definition (vgl. [Wah89]) beschreiben:

Definition: Unter der Segmentierung eines diskreten Bildsignals $f(m, n)$ mit
$(0 \leq m \leq M-1) \wedge (0 \leq n \leq N-1)$ versteht man die Unterteilung von f in disjunkte, nichtleere Teilmengen $f_1, f_2, \ldots, f_P$, so dass mit einem zu definierenden Einheitlichkeitskriterium E gilt:

(a) $\bigcup_{i=1}^{P} f_i = f$

(b) f_i ist zusammenhängend $\forall i$ mit $i = 1, \ldots, P$.

(c) $\forall f_i$ ist das Einheitlichkeitskriterium $E(f_i)$ erfüllt.

(d) Für jede Vereinigungsmenge zweier benachbarter f_i, f_j ist $E(f_i \cup f_j)$ nicht erfüllt.

Bedingung (a) verlangt, dass die Vereinigung aller segmentierten Bildelemente wieder das komplette Bild ergibt. Bedingung (b) fordert, dass die einzelnen Teilbereiche in sich zusammenhängend, also nicht quer über das Bild verteilt sind. Nach Bedingung (c) müssen die unterschiedlichen Bildbereiche ein gegebenes Einheitlichkeitskriterium erfüllen. So könnte z.B. gefordert werden, dass alle Pixel eines segmentierten Bildbereichs eine ähnliche Intensität aufweisen. Schließlich besagt Bedingung (d), dass die Pixel zweier benachbarter Teilbereiche das Einheitlichkeitskriterium nicht erfüllen dürfen. Wäre dies der Fall, müssten beide Teilbereiche gemäß der Definition zu einem einzigen zusammengefasst werden.
Bereichswachstumsverfahren (siehe Kapitel 4.2.3) sind sehr gut geeignet, um eine vollständige Segmentierung zu erzielen. Die größtenteils sehr simplen kantenorientierten Verfahren (siehe Kapitel 4.2.2) führen jedoch häufig zu Lücken oder produzieren fehlerhafte Trennungslinien innerhalb von zusammengehörigen Bildbereichen, so dass die strengen Bedingungen der vollständigen Segmentierung nicht erfüllt werden. Diese unvollständigen Segmentierungen können jedoch mit zusätzlichem Arbeitsaufwand (Konturverfolgung und Kantenelimination) in vollständige Segmentierungen überführt werden.

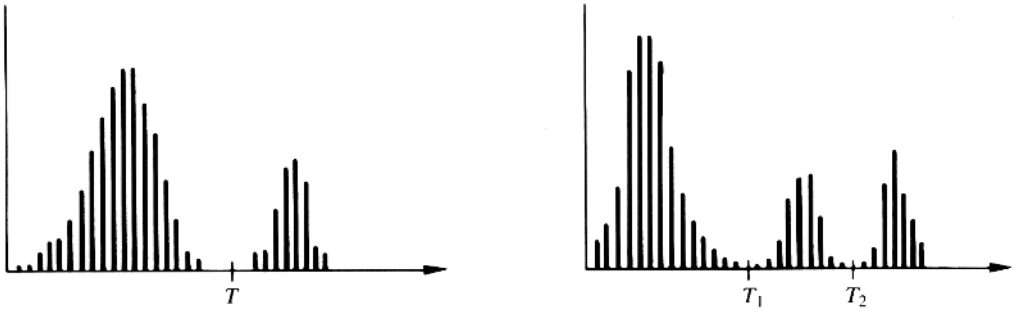


Abbildung 4.1: Histogramme mit Schwellenwerten. [GW92]

4.2 Grundlegende Segmentierungstechniken

4.2.1 Schwellenwertverfahren

Schwellenwertverfahren unterteilen Bilder ausgehend von der Intensität der einzelnen Pixel. Die Schwellenwerte begrenzen Intensitätsbereiche, welche die Zuordnung der Pixel zu unterschiedlichen Segmenten repräsentieren.

4.2.1.1 Einfache Schwellenwertverfahren

Das wohl einfachste Bildsegmentierungsverfahren stellt das einfache Schwellenwertverfahren (Simple Global Thresholding) dar. Hier wird von der idealisierten Annahme ausgegangen, dass sich die einzelnen Bildobjekte in ihrer Intensität deutlich vom Bildhintergrund abheben. Man wählt also einen geeigneteten Schwellenwert T und vergleicht dann jeden einzelnen Bildpunkt mit diesem Wert (siehe Abbildung 4.1). Liegt ein Punkt unterhalb des Schwellenwertes, wird er dem Hintergrund zugeordnet und mit dem Wert 0 markiert. Liegt er über dem Schwellenwert, wird er als Objekt (Wert 1) markiert. Man erhält so ein Binärbild, in dem Objekte und Hintergrund eindeutig voneinander getrennt sind.
Um eine feinere Unterscheidung der Bildelemente als nur die Zuordnung zu Hintergrund bzw. Objekt zu erreichen, kann es sinnvoll sein, mehrere Schwellenwerte zu definieren. Unter der Annahme, dass sich die unterschiedlichen Bildobjekte auch deutlich in ihrer Intensität voneinander unterscheiden, lassen sich im Bildhistogramm verschiedene Intensitätsbereiche ausfindig machen, die den unterschiedlich hellen Objekten entsprechen. Die lokalen Minima des Histogramms liefern somit geeignete Schwellenwerte, mit denen das Bild wie folgt aufgeteilt wird:

$$g(x,y) = \begin{cases} 0, \text{ wenn } 0 \leq f(x,y) \leq T_1 \\ 1, \text{ wenn } T_1 < f(x,y) \leq T_2 \\ \dots \\ i, \text{ wenn } T_i < f(x,y) \leq T_{i+1} \\ i+1, \text{ wenn } T_{i+1} < f(x,y) \leq T_N \end{cases} \tag{4.1}$$

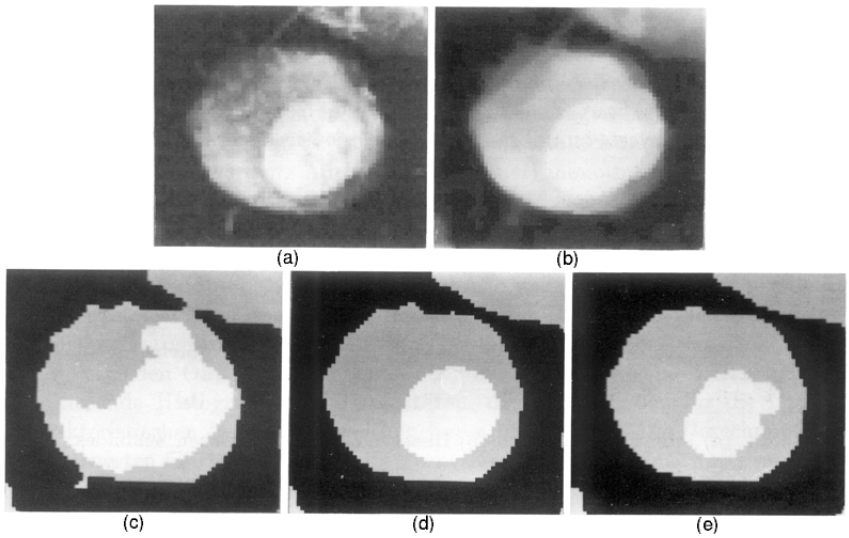


Abbildung 4.2: Einfaches Schwellenwertverfahren. (a) Originalbild einer Zelle, (b) Mediangefiltertes Bild, (c) - (d) Ergebnisse des Schwellenwertverfahrens: (c) Schwellenwerte zu niedrig gewählt, (d) Schwellenwerte optimal gewählt, (e) Schwellenwerte zu hoch gewählt. [Wah89]

Abbildung 4.2 zeigt, wie sehr die Qualität des Ergebnisses von der Wahl geeigneter Schwellenwerte abhängt. Die oberen beiden Bilder zeigen das Originalbild (Abbildung 4.2, Bild a), sowie ein mediangefiltertes Bild (b), auf welches daraufhin das Schwellenwertverfahren angewendet wurde. Das optimale Ergebnis ist in Bild d zu sehen. In den beiden anderen Bildern wurden die Schwellenwerte zu niedrig (c) bzw. zu hoch (e) gewählt. Starkes Bildrauschen und variierende Beleuchtungsverhältnisse innerhalb des Bildes können das Ergebnis dieses einfachen Verfahrens zusätzlich negativ beeinflussen.

4.2.1.2 Optimale Schwellenwertverfahren

Zur Bestimmung optimaler Schwellenwerte geht man zunächst davon aus, dass sich die Intensitätsverteilungen der einzelnen Bildobjekte durch Normalverteilungen beschreiben lassen. Das reale Histogramm des kompletten Bildes versucht man dann durch Überlagerung dieser Normalverteilungen anzunähern. Abbildung 4.3 zeigt in der oberen Reihe die Verläufe zweier immer dichter zusammengerückter Normalverteilungen. Die optimalen Schwellenwerte lassen sich problemlos als jene Intensitätswerte erkennen, an denen sich die beiden Kurven schneiden. Die untere Reihe zeigt hingegen nur das Gesamthistogramm, welches sich aus der Überlagerung der beiden Verteilungen ergibt. Hier werden die Schwellenwerte auf traditionelle Weise ermittelt, also anhand der lokalen Minima im Histogramm. Folgendes ist deutlich zu erkennen:

1. Je näher die Verteilungen zweier Bildobjekte zusammen liegen, umso mehr weicht der anhand des lokalen Minimums ermittelte Schwellenwert vom optimalen Schwellenwert ab.

2. Liegen die Einzelverteilungen zu dicht beieinander, ist eine Schwellenwertbestimmung über das lokale Minimum nicht mehr möglich.

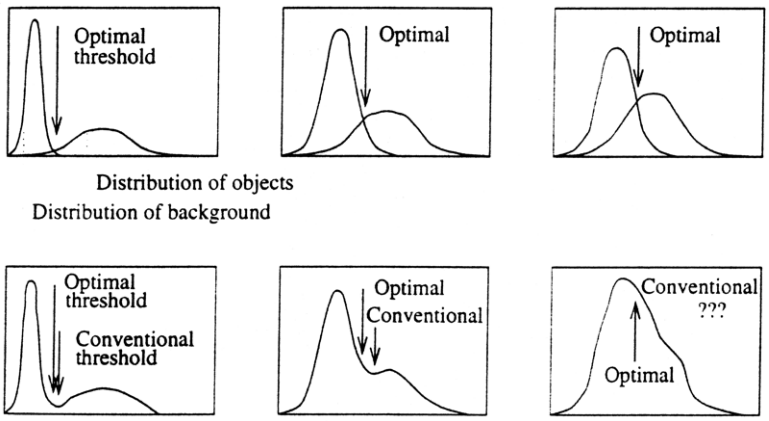


Abbildung 4.3: Optimales Schwellenwertverfahren. [Wah89]

Um nun ein komplettes Bild zu segmentieren, wird es zunächst in mehrere kleine, sich überlappende Fenster unterteilt, deren Histogramme auf Bimodalität untersucht werden. In Abbildung 4.3 sind die ersten beiden Histogramme bimodal, d.h. es lassen sich deutlich zwei unterschiedliche Intensitätsbereiche erkennen, während das dritte Histogramm unimodal ist. In den bimodalen Fenstern werden daraufhin die optimalen Schwellenwerte berechnet (durch Überlagerung von Normalverteilungen). Die Schwellenwerte der unimodalen Fenster werden aus den benachbarten Fenstern interpoliert. Danach wird durch eine weitere Interpolation eine Funktion $f(x, y)$ gebildet, die jedem einzelnen Bildpunkt einen optimalen Schwellenwert zuordnet. Schließlich wird das gesamte Bild mit dem Schwellenwertverfahren und den aus der Funktion gegebenen Schwellenwerten segmentiert.

4.2.2 Erkennung von Punkten/Linien/Kanten

4.2.2.1 Punkterkennung

Um einzelne Bildpunkte zu erkennen, die sich in ihrer Intensität stark von benachbarten Pixeln unterscheiden, faltet man das Bild mit folgender 3x3-Maske:

-1	-1	-1
-1	8	-1
-1	-1	-1

Abbildung 4.4: Faltungsmaske für Punkterkennung

Die (diskrete) Faltung arbeitet folgendermaßen:
Eine Maske wird zentriert über einen betrachteten Bildpunkt gelegt. Dann werden alle unter der Maske liegenden Punkte mit den Werten der Maske multipliziert und es wird die Summe aller so gewichteten Werte gebildet.
Diese Summe R ergibt bei gleichmäßigen Flächen den Wert 0. Je mehr sich der Mittelpunkt von seinen Nachbarn unterscheidet, umso größer wird R. Sobald R einen gegebenen Grenzwert T

übersteigt, kann man davon ausgehen, einen isolierten Bildpunkt gefunden zu haben. So wird für alle Bildpunkte nacheinander verfahren, bis sich ein Ergebnisbild ergibt, in dem die isolierten Bildpunkte deutlich erkennbar sind.
Aufgrund seiner starren Faltungsmasken eignet sich dieses Verfahren nur zur Segmentierung von Punkten mit einer festen Größe von 1 Pixel.

4.2.2.2 Linienerkennung

Die einfache Linienerkennung arbeitet nach einem ähnlichen Prinzip wie die Punkterkennung. Das Bild wird wieder gefaltet, diesmal jedoch mit 4 verschiedenen Masken, die im Folgenden dargestellt sind:

-1	-1	-1
2	2	2
-1	-1	-1

-1	-1	2
-1	2	-1
2	-1	-1

-1	2	-1
-1	2	-1
-1	2	-1

2	-1	-1
-1	2	-1
-1	-1	2

Abbildung 4.5: Faltungsmasken zur Linienerkennung

Wie leicht zu erkennen ist, liefern alle Masken auf gleichmäßigen Flächen wieder das Ergebnis 0. Besonders stark reagieren die einzelnen Masken dagegen auf durchgehende, gerade Linien. Maske 1 liefert bei horizontalen Linien das größte Ergebnis, während z.B. Maske 2 besonders auf um 45° gedrehte Linien reagiert. Da das Bild mit allen vier Masken gefaltet wird, kann man für jeden Bildpunkt von der am stärksten reagierenden Maske auf die Orientierung der Linie im Bild schließen.
Diese Methode hat ebenfalls den Nachteil, dass sie aufgrund der starren Faltungsmasken nur für Linien von exakt 1 Punkt Breite geeignet ist.

4.2.2.3 Kantenerkennung

Zur Identifizierung von Bildkanten bieten sich die 1. und 2. Ableitung der Bildfunktion an. Abbildung 4.6 soll die Bedeutung der beiden Ableitungen für die Bildauswertung veranschaulichen. In der ersten Zeile sind zwei einfache Bilder dargestellt, die einen hellen bzw. dunklen vertikalen Streifen beinhalten. Darunter sind die jeweiligen Intensitätsverläufe der Bildpunkte entlang einer einzelnen Bildzeile aufgezeichnet. In der dritten Zeile ist der Verlauf der ersten Ableitung dieser Intensitätskurve dargestellt. Es ist zu erkennen, dass die erste Ableitung nur an den Stellen von 0 abweicht, an denen im Ausgangsbild ein deutlicher Intensitätswechsel erfolgt. Der Betrag der ersten Ableitung ist also offenbar ein guter Indikator für die Existenz einer Kante. Im ganz unten dargestellten Verlauf der 2. Ableitung lassen sich gleich zwei Merkmale erkennen. Zum einen ermöglicht das Vorzeichen eine eindeutige Zuordnung eines Kantenpixels zur hellen oder dunklen Seite der Kante. Zum anderen kann man anhand der 2. Ableitung auch die exakte Position einer Kante bestimmen, da sie genau an dieser Stelle einen Nulldurchgang aufweist. Das ist besonders bei unscharfen Kanten von Vorteil, doch dazu mehr am Ende dieses Unterkapitels.
Die 1. Ableitung lässt sich durch den Gradientenvektor darstellen:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{4.2}$$

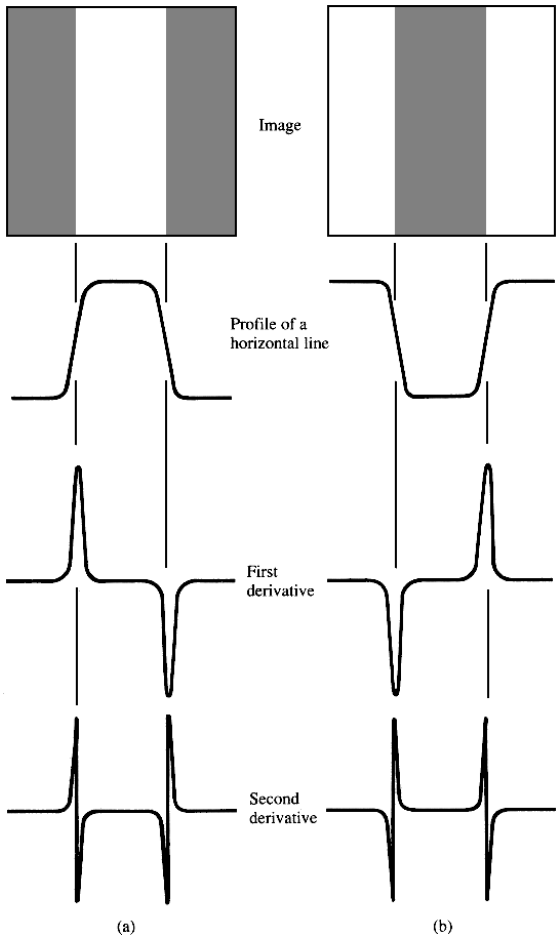


Abbildung 4.6: Bedeutung der ersten und zweiten Ableitung. (a) Heller Streifen auf dunklem Hintergrund, (b) Dunkler Streifen auf hellem Hintergrund. Zu sehen sind jeweils das Originalbild, der Intensitätsverlauf entlang einer Zeile, sowie die erste und zweite Ableitung dieses Verlaufs. [GW92]

Er setzt sich aus den partiellen Ableitungen der Bildfunktion in x- und y-Richtung zusammen. Der Gradientenvektor zeigt also für jeden Bildpunkt in Richtung der maximalen Intensitätsänderung, seine Länge gibt die Stärke dieser Änderung an. Der Vektor steht senkrecht auf den im Bild wahrgenommenen Kanten.

Die Berechnung des Gradientenvektors erfolgt wieder über diskrete Faltung des Bildes mit geeigneten Masken. So berechnen die in Abbildung 4.7 dargestellten Masken die partiellen Ableitungen in x- und y-Richtung.

Diese einfachen Masken sind jedoch sehr rauschanfällig, weshalb man üblicherweise Faltungsmasken benutzt, die gleichzeitig eine leichte Bildglättung beinhalten. Ein beliebter Kantendetektor ist z.B. der Sobel-Operator, der den Gradientenvektor aus den beiden in Abbildung 4.8

-1
0
1

-1	0	1

Abbildung 4.7: Einfache Faltungsmasken zur Gradientenberechnung

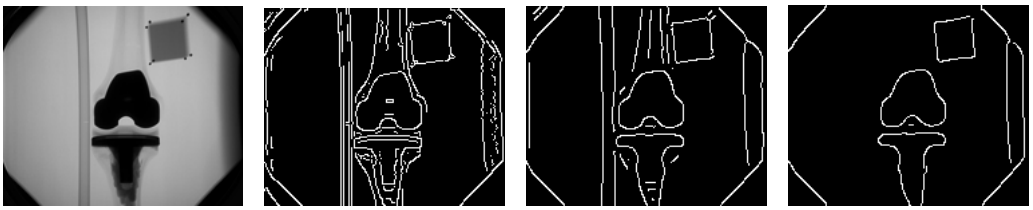
gezeigten Masken ermittelt.

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Abbildung 4.8: Faltungsmasken des Sobel-Operators

Neben den simplen Faltungsoperatoren soll nun noch auf einen etwas geschickteren Kantendetektor eingegangen werden, den Canny-Operator (Anwendungsbeispiele siehe Abbildung 4.9).



(a) Originalbild (b) $\sigma = 0.4$ (c) $\sigma = 2.0$ (d) $\sigma = 4.0$

Abbildung 4.9: Der Canny-Operator

Der Canny-Operator glättet zunächst das Bild mittels Gauß-Funktion (siehe Kapitel 4.4.1.3). Die Standardabweichung σ ist einer der Eingabeparameter des Algorithmus und bestimmt die Stärke der Glättung. Von dem geglätteten Bild wird dann die erste Ableitung berechnet, man erhält ein Gradientenbild.
Der nächste Schritt ist die Kantenverdünnung. Für jeden Bildpunkt wird zunächst angenommen, dass er auf einer Kante liegt. Da Gradientenvektoren stets senkrecht auf einer Kante stehen, vergleicht man den aktuellen Punkt mit denen, die in der Nachbarschaft entlang der Richtung des Gradienten liegen. Ist der Gradient eines Nachbarpunktes größer, kann der aktuell betrachtete Punkt kein Kantenpunkt sein und wird somit auf Null gesetzt. Es bleiben nur die Bildpunkte übrig, deren Gradienten lokale Maxima sind. Abschließend wird das Bild mit einem speziellen Schwellenwertverfahren („Hysteresis Thresholding“) bearbeitet. Dieses arbeitet in zwei Schritten. Zunächst werden mit einem relativ hohen Schwellenwert nur die Punkte markiert, die definitiv zu Bildkanten gehören. Von diesen Punkten ausgehend werden dann mit einem zweiten, niedrigeren Schwellenwert die restlichen Kantenpunkte markiert. So soll sichergestellt werden, dass in den segmentierten Kanten keine Lücken entstehen.

Die zweite Ableitung lässt sich mit dem sogenannten Laplace-Operator berechnen:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{4.3}$$

Folgende Faltungsmaske ist zur Berechnung der zweiten Ableitung weit verbreitet:

0	-1	0
-1	4	-1
0	-1	0

Abbildung 4.10: Faltungsmaske des Laplace-Operators

Allerdings ist der Laplace-Operator an sich sehr rauschanfällig und wird deshalb meist nur in Kombination mit anderen Filtern benutzt.
Das folgende Verfahren nutzt die zu Beginn dieses Unterkapitels erwähnten Haupteigenschaften der zweiten Ableitung (Pixelzuordnung zur korrekten Kantenseite über das Vorzeichen, Bestimmung der exakten Kantenmitte durch Nulldurchgänge) und ist geeignet, selbst aus unscharfen Bildern eindeutige, scharfe Kanten zu segmentieren. An Abbildung 4.11 lässt sich der Algorithmus recht gut nachvollziehen.
Das Originalbild wird zunächst mit einem Gauß-Filter geglättet. Danach wird das Bild mit dem empfindlichen Laplace-Operator gefaltet. Das Ergebnis ist in Bild (b) zu sehen: Der Kontrast sämtlicher Kanten wurde stark verbessert, auch die weichen Übergänge weisen nun relativ scharfe Hell-Dunkel-Wechsel auf. Es sei noch angemerkt, dass der Wertebereich des Laplace-Operators um 50% verschoben wurde, um alle Ergebnisse mit positiven Grauwerten darstellen zu können. Ein mittlerer Grauwert steht also für eine Null in der zweiten Ableitung, hellere Grauwerte stehen für eine positive zweite Ableitung, während dunkle Grauwerte die negative zweite Ableitung darstellen. Um die Nulldurchgänge (und damit die exakten Kantenpositionen) sichtbar zu machen, wird das Bild im nächsten Schritt in ein Binärbild umgewandelt. Punkte, deren zweite Ableitung unter Null liegt, werden im Binärbild auf Schwarz gesetzt, Punkte mit positiver zweiter Ableitung auf Weiß. Aus diesem Bild lassen sich nun leicht die Nulldurchgänge ermitteln, als Grenzen zwischen schwarzen und weißen Regionen.

Vergleich Gradient/Laplace-Operator:

- Der Gradient ist sehr gut zur Kantendetektion in rauscharmen Bildern mit scharfen Kanten geeignet.

- Der Laplace-Operator eignet sich (vorherige Bildglättung vorausgesetzt) besonders zur Detektion von unscharfen Kanten.

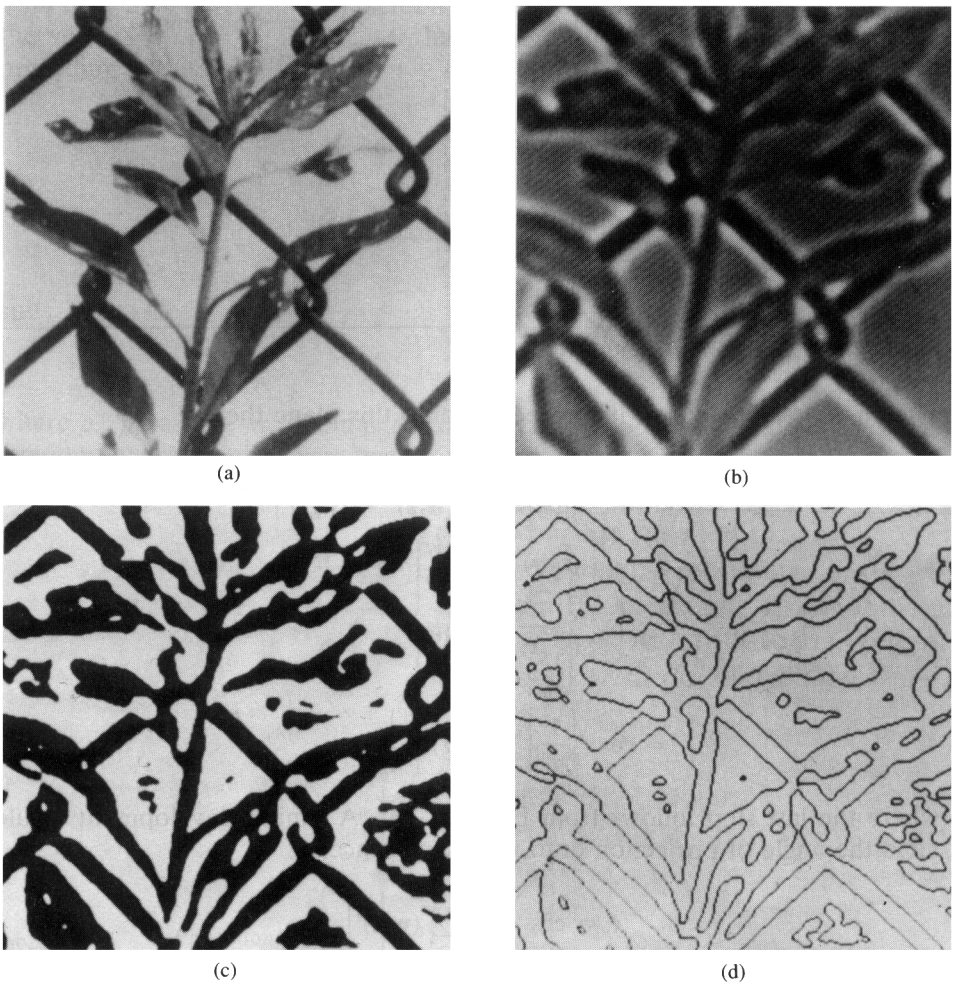


Abbildung 4.11: Kantenberechnung mit Hilfe von Nulldurchgängen. (a) Originalbild, (b) Gaußglättung und Faltung mit Laplace-Operator, (c) aus (b) erzeugtes Binärbild, (d) die gefundenen Kanten (Nulldurchgänge) [GW92]

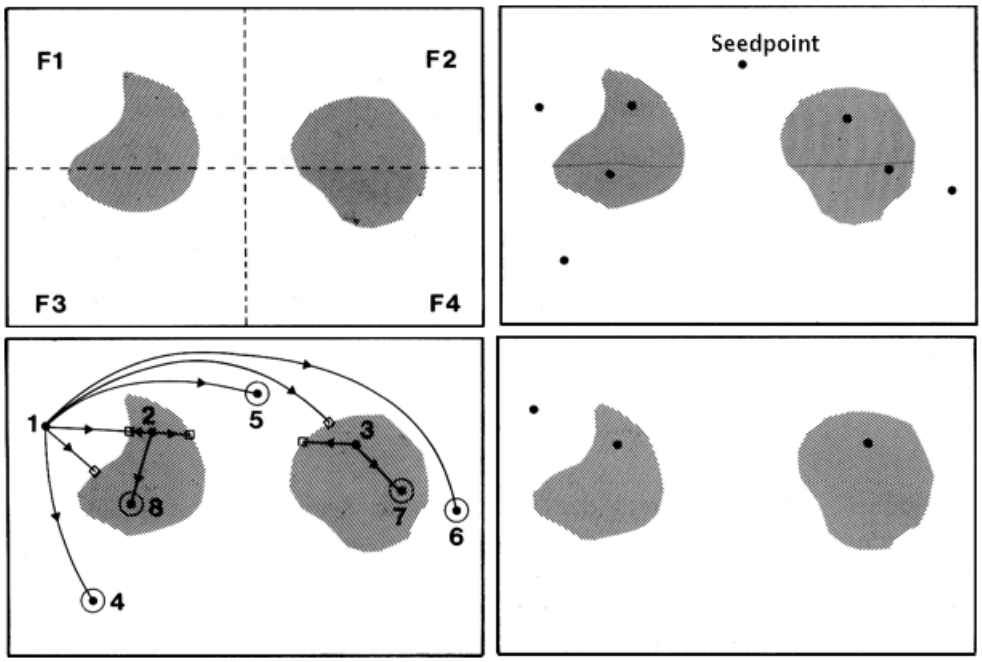


Abbildung 4.12: Bestimmung der Seedpoints des Bereichswachstumsverfahrens. (a) Unterteilung des Bildes in mehrere Fenster, (b) Markierung potentiell geeigneter Punkte, (c) Eliminierung überflüssiger Punkte, (d) Gefundene Seedpoints für das Bereichswachstumsverfahren. [Wah89]

4.2.3 Bereichswachstumsverfahren

Das klassische Bereichswachstumsverfahren eignet sich besonders, um eine vollständige Segmentierung mit zusammenhängenden Unterbereichen zu erzielen. Ausgehend von geeigneten Startpunkten - den so genannten Seedpoints, die unterschiedlichen Teilmengen zugeordnet sind - wird ein iterativer Prozess gestartet. Die Punkte in direkter Nachbarschaft zu den Anfangs- oder weiteren bereits zugeordneten Punkten werden nach sinnvollen Kriterien (üblicherweise ihrer Intensität) bewertet. Ist die Differenz zum Mittelwert des bereits segmentierten Bereichs nicht zu groß (überschreitet also nicht den vorgegebenen Schwellenwert), werden die entsprechenden Punkte auch diesem Bereich zugewiesen.
So wachsen um die Seedpoints herum die verschiedenen Teilbereiche des Bildes, bis das komplette Bild segmentiert ist. Sollten ab einem gewissen Iterationsschritt keine neuen Punkte mehr einem Bereich zugewiesen werden können, obwohl das Bild noch nicht komplett segmentiert wurde, erhöht man schrittweise den Schwellenwert, bis schließlich alle Bildpunkte irgendeinem Teilbereich angehören.
Da man nicht davon ausgehen kann, dass die Seedpoints ideal gewählt wurden (exakt 1 Punkt pro tatsächlichem Teilbereich im Bild), müssen anschließend noch nebeneinanderliegende Bereiche mit gleichen Merkmalen zusammengefasst werden.

Abbildung 4.12 zeigt ein mögliches Verfahren zur Bestimmung der Seedpoints.
Das Bild wird zunächst in mehrere Fenster unterteilt, um Störungen durch unterschiedliche Beleuchtungsverhältnisse zu mindern (a). Dann werden Punkte markiert, deren Intensitätsgradien-

ten einen festgelegten Wert nicht überschreiten (b). Dadurch wird verhindert, dass Punkte ausgewählt werden, die z.B. genau auf einer Kante liegen und somit für das Bereichswachstumsverfahren ungeeignet sind.
Anschließend sollten überflüssige Punkte eliminiert werden. Dazu wird überprüft, ob der jeweils betrachtete Punkt von einem anderen aus zu erreichen ist, ohne auf der entsprechenden Wegstrecke einen zu großen Intensitätsgradienten (also eine Objektgrenze) überqueren zu müssen (c). Da es zu aufwendig wäre, alle möglichen Wege zwischen zwei Punkten zu untersuchen, beschränkt man sich in der Regel auf die simple Verbindung per gerader Linie und untersucht nur die Punkte, die auf dieser Gerade liegen.
Bild d zeigt schließlich eine ideale Auswahl von Seedpoints.

4.2.4 Hough-Transformation

Die Hough-Transformation eignet sich besonders zur Erkennung einfacher geometrischer Formen, wie z.B. Geraden oder Kreise. Dazu muss zunächst eine geeignete Parameterdarstellung, bestehend aus n Parametern, gefunden werden, so dass jedes n-Tupel eindeutig die genaue Lage und Orientierung eines gesuchten Objekts im Bild repräsentiert. So lassen sich z.B. Geraden durch die Hesse'sche Normalenform mit den Parametern Winkel und Länge eines Normalenvektors beschreiben (siehe Kapitel 4.2.4.1), zur eindeutigen Zuordnung eines Kreises bieten sich Position (x und y), sowie der Kreisradius an (siehe Kapitel 4.2.4.2). Aus diesen Parametern wird dann ein n-dimensionaler Hough-Raum aufgespannt, der als Zähler für die folgende Transformation dient. Jede Zelle des Hough-Raums stellt durch ihre Koordinaten potentiell im Bild vorhandene Objekte eindeutig dar.
Bei der Hough-Transformation wird nun das Ausgangsbild (in der Regel ein Gradientenbild) Pixel für Pixel durchlaufen. Jedes gesetzte Pixel könnte nun Teil mehrerer unterschiedlich positionierter Suchobjekte sein. Das oberste Pixel eines Kreises könnte genauso gut das unterste Pixel eines Kreises mit gleichem Radius r sein, der sich aber in y-Richtung um 2r nach oben verschoben befindet. Also werden im Hough-Raum die Inhalte aller Zellen um 1 erhöht, zu deren repräsentierten Objekten das aktuelle Pixel gehören könnte. Dadurch entstehen im Hough-Raum genau an den Stellen Maxima, an denen es die größte Übereinstimmung zwischen gesuchtem Objekt und Ausgangsbild gibt.
Die Hough-Transformation ist relativ unanfällig für lokale Störungen, solange sich trotz der Unterbrechungen der Objektkonturen im Ausgangsbild eindeutige Maxima im Hough-Raum ausbilden.

4.2.4.1 Linienerkennung mittels Hough-Transformation

Eine geeignete Parameterdarstellung der Geraden ist die Hesse'sche Normalenform. Diese beschreibt einen im Ursprung des Koordinatensystems liegenden Normalenvektor mit Winkel ϕ und Länge s, an dessen Spitze die Gerade senkrecht anliegt (siehe Abbildung 4.13).
Die Geradengleichung lautet:

$$x \cos \phi + y \sin \phi - s = 0 \tag{4.4}$$

Nun durchläuft man für jedes gesetzte Pixel den Winkelbereich, in dem nach Geraden gesucht wird, und berechnet die zugehörige Länge s des Normalenvektors. Der resultierende Hough-Raum ergibt ein Sinogramm mit Maxima an den Stellen, auf deren Geraden sich im Bild die meisten Pixel befinden.

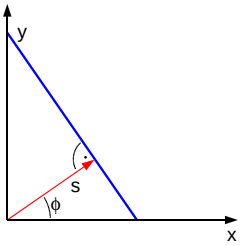


Abbildung 4.13: Geradenbeschreibung durch Hesse'sche Normalenform.

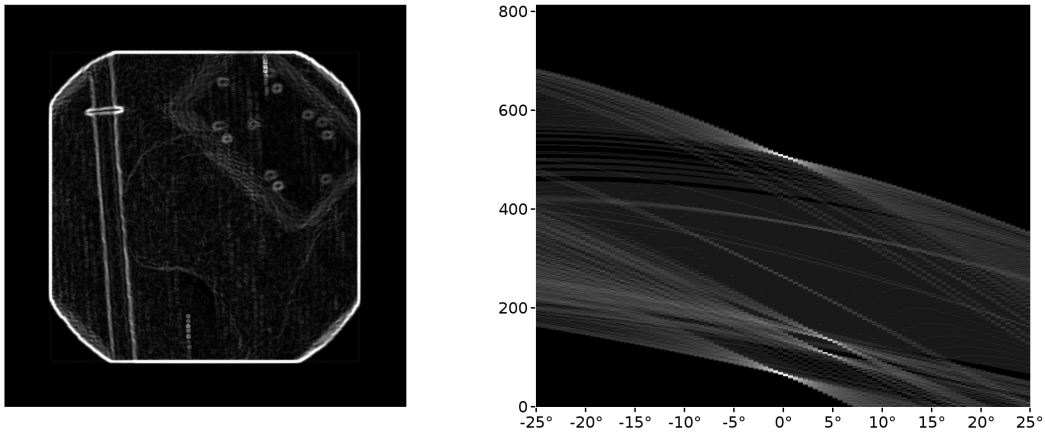


Abbildung 4.14: Linien-Hough-Transformation. Links ein Kantenbild, rechts das dazugehörige Sinogramm der Hough-Transformation.

Abbildung 4.14 zeigt ein Kantenbild und die Hough-Transformation im Bereich -25° bis 25°, also für vertikale Linien. Deutlich erkennbar sind die beiden Maxima bei 0°, die von den starken Kanten am Bildrand stammen, sowie die beiden dicht zusammenliegenden Maxima bei 5°, die durch den Stab entstanden sind.

4.2.4.2 Kreiserkennung mittels Hough-Transformation

Die Kreiserkennung funktioniert ähnlich der Geradenerkennung, allerdings wird der Hough-Raum hier dreidimensional. Neben der eindeutigen Position (x, y) muss nämlich zusätzlich der Kreisradius berücksichtigt werden. Dieser sollte aus Effizienzgründen stark eingegrenzt werden. Somit eignet sich die Hough-Transformation eigentlich nur zur Kreissuche, wenn die Größe der Kreise im Ausgangsbild nicht allzu stark variiert.
Im Verlauf der Transformation werden einfach alle Punkte, die um einen Kantenpunkt (x, y) einen Kreis mit Radius r bilden, im Hough-Raum markiert. Werden mehrere Radien zugelassen, erhält jeder Radius seine eigene zweidimensionale Tabelle, in der nur Kreise mit dem jeweiligen Radius markiert werden.
Die Kreisgleichung eines Kreises mit Radius r und Mittelpunkt $(x_0,\ y_0)$ lautet:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \tag{4.5}$$

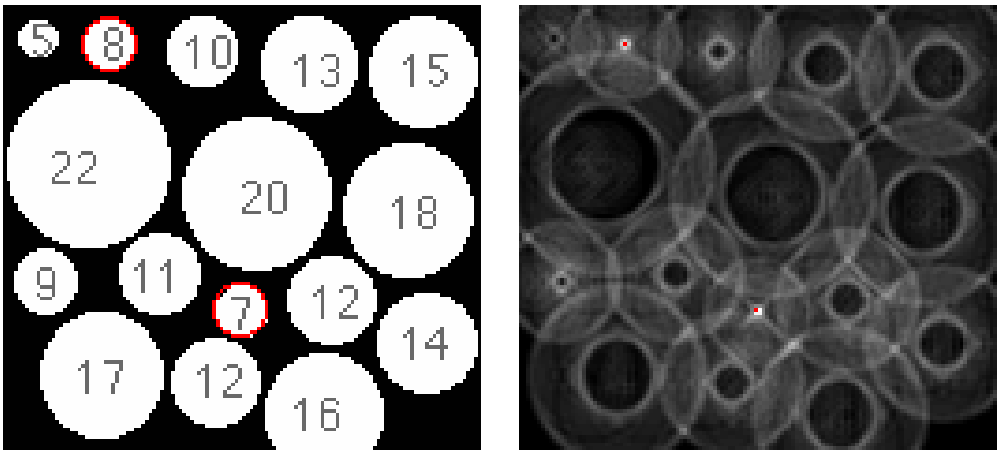


Abbildung 4.15: Kreis-Hough-Transformation. Links das Ursprungsbild, rechts der abgebildete Houghraum. [Sch03]

Auch mit diesem Verfahren ergeben sich im Hough-Raum wieder einzelne Maxima, die einen Kreis bestimmter Größe an einer bestimmten Position im Ausgangsbild repräsentieren.
Abbildung 4.15 zeigt ein Bild mit Kreisen unterschiedlicher Radien, sowie den aus dessen Gradientenbild abgeleiteten Houghraum für den Radius 7. Gut zu erkennen sind die 14 Pixel breiten Ringe, die sich um die tatsächlichen Kreiskanten herum bilden. Stimmt der Radius eines Kreises im Bild mit dem Suchradius überein, liegt die innere „Kante“ des entsprechenden Rings im Houghraum genau im Mittelpunkt des Kreises und bildet dort ein starkes Maximum aus. Im Bild wurde außerdem ein zweites (schwächeres) Maximum erkannt und markiert: Das eines Kreises mit Radius 8.

4.3 Modellbasierte Verfahren

Die bisher vorgestellten Segmentierungsverfahren haben einen entscheidenden Nachteil: Ein Benutzer, der schon genaue Vorstellungen von dem zu segmentierenden Objekt hat, kann dieses a priori-Wissen nicht direkt in den Segmentierungsprozess einbringen. Die bisherigen Verfahren bearbeiten immer das gesamte Bild und segmentieren somit auch unbedeutende, schlimmstenfalls fehlerhafte Elemente heraus.
Genau hier setzen die modellbasierten Verfahren an. Der Benutzer kann ein Modell des zu segmentierenden Objektes vorgeben, und der Algorithmus sucht dann im Bild gezielt nach diesem Modell und versucht, es in das Bild einzupassen.

4.3.1 Statistisches Modell zur Formbeschreibung

Das hier zu besprechende Modell enthält globale Bedingungen für die Form. Diese werden anhand von Trainingsbildern gelernt und enthalten durch eine passende Beschreibung genügend Spielraum, um plausible und legale Instanzen des Modells zu erstellen. Zur Segmentierung erfolgt ein Matching von passenden Modellinstanzen mit dem Bildinhalt. Dieses Modell heißt *Active Shape Model*, kurz *ASM*.

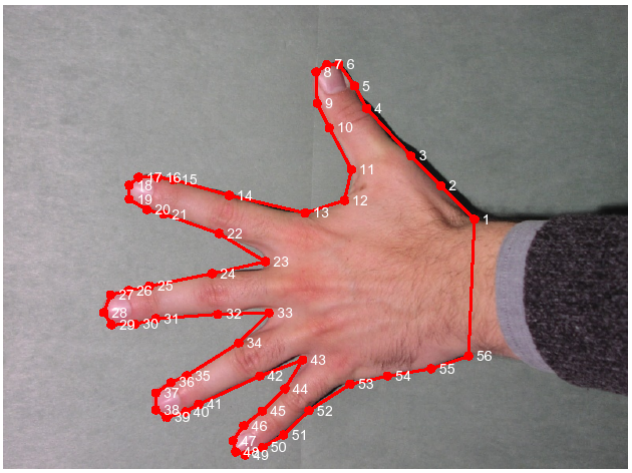


Abbildung 4.16: Beispiel: weit gespreizte Finger [SG01]

Zunächst werden die Stationen der Modellbildung beschrieben, danach der eigentliche Segmentierungsvorgang. Weitere Details finden sich in [PG 04].

4.3.1.1 Beschreiben der Form

Die nachfolgenden Definitionen folgen der Einführung zu *Active Shape Models* in [SG01]. Zunächst soll erläutert werden, was unter dem Begriff Form zu verstehen ist. Man kann die Form eines Objekts als die gesamte geometrische Information bezeichnen, welche übrig bleibt, wenn Positions-, Skalierungs- und Rotationseffekte herausgefiltert werden. Dieser Definition folgend ist Form also zu euklidischen Ähnlichkeitstransformationen invariant - zumindest sollte sie es sein. Ob dieses Versprechen eingehalten werden kann, hängt von der Modellierung ab.
Eine naheliegende Möglichkeit zur Beschreibung der Form besteht darin, den Umriss eines Objekts durch eine endliche Menge an Punkten zu beschreiben. Diese Punkte werden Landmarken genannt. Diese Landmarken sind Korrespondenzpunkte, welche so in jeder Objektbeschreibung d.h. in jedem einzelnen Trainingsbild vorkommen. Dabei können Landmarken anatomischer, mathematischer oder einfach nur füllender Natur sein (Details in [CT01a], [SG01]).
Jetzt gilt es, eine mathematische Repräsentation zu definieren. Eine Form, welche aus n Punkten in d Dimensionen besteht, kann zu einem nd-dimensionalen Vektor zusammengefasst werden. Für den 2-dimensionalen Fall hat der Formvektor somit folgendes Aussehen:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n]^T \tag{4.6}$$

Dieser Vektor **x** wird für jedes Trainingsbild aufgestellt. Bei s Bildern demnach s Vektoren mit je nd Einträgen bzw. Dimensionen. Im Nachfolgenden gilt stets $d = 2$, s gibt den Umfang der Trainingsmenge an.
Die beiden Bilder 4.16 und 4.17 geben eine Vorstellung davon, wie so eine Beschreibung aussehen kann. Die abgebildeten Hände wurden mit jeweils 56 Landmarken (112 Dimensionen!) beschrieben und zeigen zwei extreme Fingerstellungen. Natürlich umfasst die Trainingsmenge mehr als nur diese beiden Bilder - Varianten zwischen diesen Extrema sowie zufällige Fingerstellungen. Mehr zu diesem Experiment in [SG01].

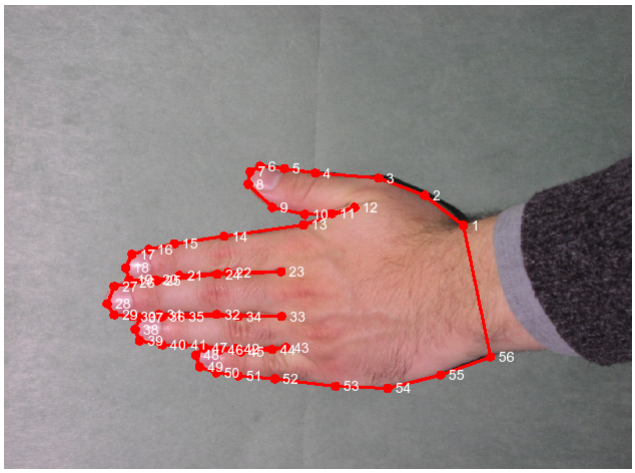


Abbildung 4.17: Beispiel: Finger zusammen [SG01]

Noch einige Anmerkungen zur Formbeschreibung: Die Form ist immer nur so gut wie der Experte, der sie modelliert hat. Es gibt in der Literatur verschiedene Ansätze, die Modellbildung zu verbessern. Der wünschenswerteste ist sicherlich die automatische Formbeschreibung. Leider ist dies schwer. Daher muss der Experte bei der Eingabe sorgfältig arbeiten. Man kann ihn unterstützen oder überprüfen, wenn man Bilder mehrmals in unterschiedlicher Reihenfolge markieren lässt.

4.3.1.2 Ausrichten der Formen

Nachdem beschrieben wurde, wie Form modelliert werden kann, wird im Folgenden dargestellt, wie man diese untereinander vergleichbar macht, d.h. in ein gemeinsames Koordinatensystem (Form-Raum) überführt. Das hierzu benutzte Verfahren wird Procrustes-Analyse genannt. Nachfolgend wird die generalisierte Procrustes-Analyse vorgestellt, welche einen iterativen anstelle eines analytischen Ansatzes benutzt. Ziel ist die Minimierung von $D = \sum |\mathbf{x_i} - \bar{\mathbf{x}}|^2$ mit $\mathbf{x_i}$ Form und $\bar{\mathbf{x}}$ Mittelwert(-Form). Es wird also versucht, den Abstand aller Formen zur Durchschnittsform zu minimieren (Fehlerquadratmethode).

Algorithmus 1 Procrustes-Analyse

1: Verschiebe jedes Beispiel, sodass sein Schwerpunkt im Ursprung liegt
2: Ein beliebiges Beispiel wird erster Mittelwert
3: Normiere Vektor $|\bar{\mathbf{x}}| = 1$
4: Richte alle verbleibenden Formen am aktuellen Mittelwert aus
5: Berechne neuen Mittelwert $\bar{\mathbf{x}} = \sum_{i=1}^{s} \mathbf{x_i}$
6: Normiere neuen Mittelwert und richte diesen am alten Mittelwert aus
7: Wenn noch nicht konvergiert, gehe zu 4

Die Konvergenz von Algorithmus 1 wird als gegeben angenommen, wenn sich alter und neuer Mittelwert nicht mehr signifikant unterscheiden.
Im oben erwähnten Hand-Beispiel handelt es sich nun um die in Abbildung 4.18 dargestellte Situation. Die vorher chaotisch anmutenden Punktpositionen haben sich an gewissen Stellen ver-

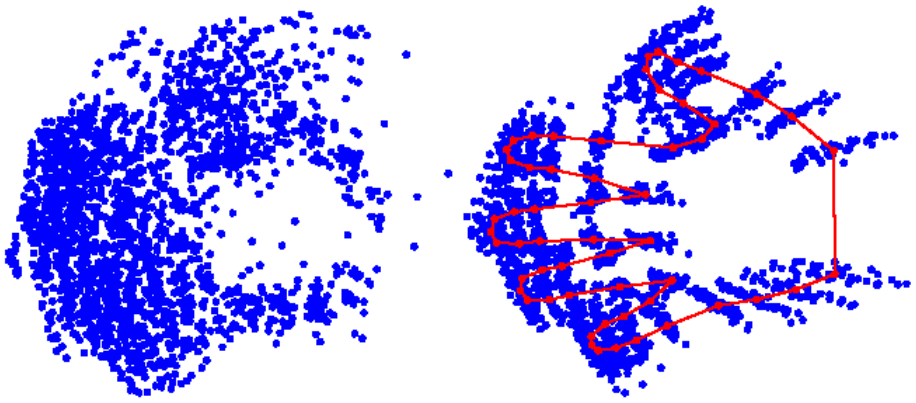


Abbildung 4.18: Ausgerichtete Formen in Punktwolken-Darstellung [SG01]

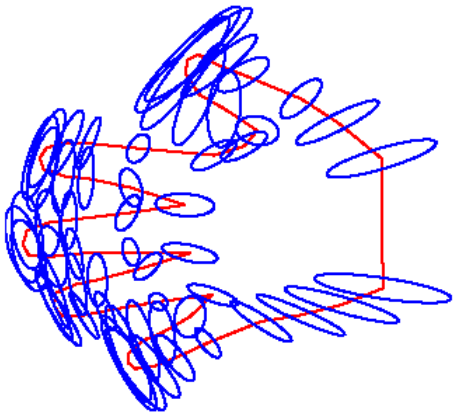


Abbildung 4.19: Ausgerichtete Formen in Ellipsoid-Darstellung [SG01]

dichtet. Diese Formationen werden Punktwolken genannt.
Noch eine Anmerkung: Bei der Procrustes-Analyse entstehen durch den Ausrichtungsprozess Nicht-Linearitäten. Diese werden durch die Projektion in den Tangentenraum von $\bar{\mathbf{x}}$ eliminiert. Diese Projektion wird durch Skalierung der restlichen Vektoren linearisiert. Weitere Details finden sich in [CT01a] oder [SG01].

4.3.1.3 Formänderungen modellieren

Es existieren nun s ausgerichtete Formen $\mathbf{x_j}$. Zur Verdeutlichung in Abbildung 4.19 noch einmal die ausgerichteten Formen in Ellipsoid- anstelle der Punktwolkendarstellung: Diese Darstellung zeigt, in welchen Regionen sich die Landmarken in den verschiedenen Trainingsbildern aufhalten. Diese Verteilung gilt es zu Modellieren.
In diesem Abschnitt wird beschrieben, wie sich Formvariationen durch lineare Algebra beschreiben lassen. Das hierzu benutzte Verfahren heißt *Pricipal Component Analysis* (PCA), welche auch als Karhunen-Loéve-, Hauptachsen- oder Eigenvektor-Transformation bekannt ist (siehe

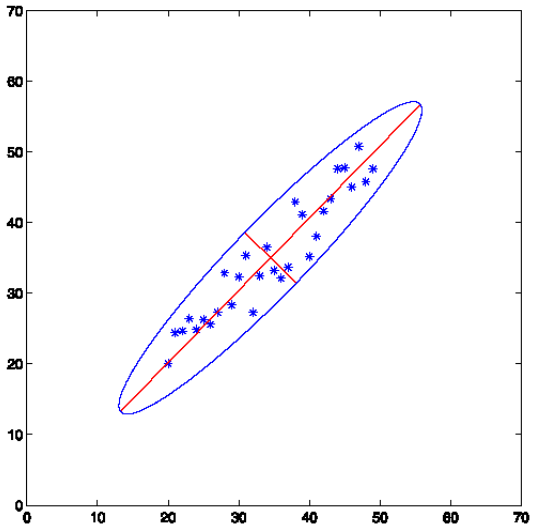


Abbildung 4.20: Beispiel: 2D-PCA - Die Geraden entsprechen den Hauptachsen , die Punkte den Landmarken [SG01]

[B$^+$00]).

Graphisch kann man sich die Funktionsweise der PCA wohl am besten verdeutlichen - siehe dazu auch Abbildung 4.20. Es werden die Hauptachsen der Punktwolken ausgerechnet, die Achsen mit dem größten Einfluss auf die Position werden beibehalten, der Rest wird eliminiert. Abbildung 4.21 zeigt die Situation nach Verwerfen der zweiten Achse. Der Punkt **x** wird durch den Punkt auf der Hauptachse **x'** approximiert. Eine mathematische Herleitung der PCA findet sich in [SG01].

Nachdem erläutert wurde, wie die Beschreibungsweise der Punktwolken generell komprimiert werden kann, wird nun das Modell näher vorgestellt: Gesucht wird eine möglichst kompakte lineare Beschreibungsweise aller nun ausgerichtet vorliegender Formen. Darüber hinaus soll das Modell so allgemein sein, dass alte sowie neue (aber plausible und legale) Modellinstanzen erzeugt werden können. Im erwähnten Beispiel dürfen demnach keine extremen Verrenkungen der Finger oder ähnliches auftauchen.

Die Aufgabe besteht also darin, ein parametrisiertes Modell $\mathbf{x} = \mathbf{M}(\mathbf{b})$ mit Parametervektor $\mathbf{b}$ zu finden. Die Parameterverteilung $p(\mathbf{b})$ ist so zu limitieren, dass die generierte Form $\mathbf{x}$ zu den Trainingsbildern ähnlich bleibt. Das große Problem dabei ist, dass mit nd Dimensionen hantiert wird. Genau dort setzt die PCA an. Es wird angenommen, dass die Formbeschreibung Redundanz enthält. Aufgrund dieser Redundanz müsste eine kompaktere Beschreibung möglich sein. Im Hand-Beispiel können sich z.B. die Landmarken eines Fingers bei anderer Fingerstellung nur komplett, aber unterschiedlich weit, verschieben. Weiterhin ist man vielleicht bereit einen gewissen Informationsverlust zuzulassen, weil man ihn z.B. als Rauschen ansieht (vgl. Einfluss der Hauptachsen).

Nach Anwenden von Algorithmus 2 kann jedes Trainingsbild durch $\mathbf{x} \approx \bar{\mathbf{x}} + \mathbf{\Phi b}$ approximiert werden. Der Vorteil liegt in der nun kleineren Dimension ($t \leq nd$) von **b**. Mit dem Vektor **b** kann nun die Form variiert werden. Die Varianz jedes Eintrags b_i von **b** wird durch den dazugehörigen Eigenwert λ_i gegeben. Limitiert man die b_i auf Werte $\pm 3\sqrt{\lambda_i}$ können Formen generiert

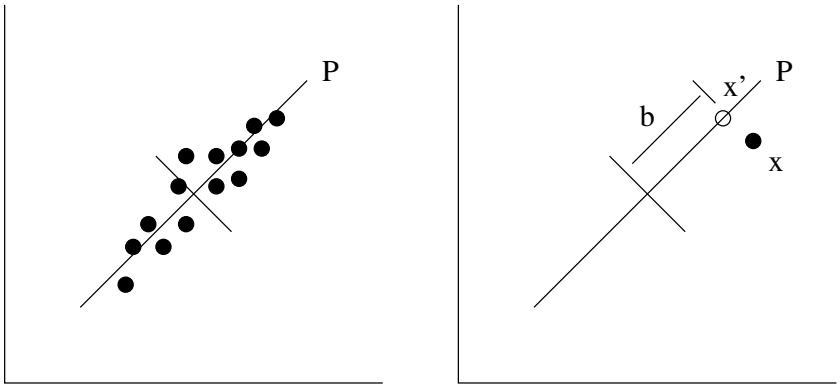


Abbildung 4.21: Beispiel: 2D-PCA - Approximation

Algorithmus 2 Principal Component Analysis (PCA)

1: Berechne den Mittelwert $\bar{\mathbf{x}} = \sum_{i=1}^{s} \mathbf{x_i}$
2: Berechne die Kovarianzmatrix der Daten: $\mathbf{S} = \frac{1}{s-1} \sum_{i=1}^{s} (\mathbf{x_i} - \bar{\mathbf{x}})(\mathbf{x_i} - \bar{\mathbf{x}})^T$
3: Berechne die Eigenvektoren $\phi_\mathbf{i}$ und die dazugehörigen Eigenwerte λ_i von $\mathbf{S}$. Sortiere $\phi_\mathbf{i}$ nach den dazugehörigen Eigenwerten, sodass $\lambda_i \geq \lambda_{i+1}$. Fasse t Eigenvektoren mit den größten Eigenwerten in einer Matrix $\mathbf{\Phi} = (\phi_\mathbf{1}|\phi_\mathbf{2}|\ldots|\phi_\mathbf{t})$ zusammen.

werden, die den Trainingsbildern ähneln ohne zu entarten. Der Wert t wird gerade so groß gewählt, dass eine gewünschte Menge der Vorlage durch Variation des Parameters b generierbar ist. Damit ist die Stärke des Informationsverlustes wählbar. Die Anzahl der Eigenvektoren t gibt die Anzahl der Moden an. Die Eigenvektoren entsprechen einer Menge von Verschiebungsvektoren (vgl. Formvektor = Menge von Punkten, vgl. Abbildung 4.21). Der Parameter **b** kontrolliert anschaulich die Längen der Vektoren.
Zurück zum Handbeispiel: Die in Abbildung 4.22 dargestellte Korrelationsmatrix, welche aus der Kovarianzmatrix errechnet wurde, zeigt Abhängigkeiten zwischen den Punkten an. Schwarze und weiße Einträge entsprechen großen Abhängigkeiten, grau geringen Abhängigkeiten.
Es folgt in Abbildung 4.24 ein Beispiel für die Variationsmöglichkeiten mit den ersten drei Einträgen im Parametervektor **b**. Die erste Mode (a,b,c) beschreibt die Spreizung der Finger, aber auch die Größe der Handfläche. Mode Nummer zwei (d,e,f) beschreibt eine ungleichmäßige Spreizung der Finger. Die dritte Mode (g,h,i) schließlich beschreibt zum einen die Spreizung von Zeige- und Mittel- sowie Ringfinger und kleinem Finger zum anderen. Im diesem Experiment erklären die ersten drei Moden 92% der Variationen, die ersten fünf Parameter erklären 96% aller Variationen (siehe Balkengrafik in Abbildung 4.23). Natürlich kommt man nicht immer mit so wenigen Einträgen im Parametervektor aus. Im Allgemeinen lässt sich aber die Dimension deutlich reduzieren.

4.3.1.4 Anpassen eines Modells an neue Punkte

Um mit dem Formmodell eine zu einem Vektor $\mathbf{Y}$ zusammengefasste neue Punktmenge nachzubilden, müssen die folgenden Parameter bestimmt werden: $\mathbf{b}, X_t, Y_t, \theta, s$. Dabei ist $\mathbf{b}$ der bekannte Parametervektor. X_t, Y_t sind Translationsparameter, θ ein Rotationswinkel und s ein Ska-

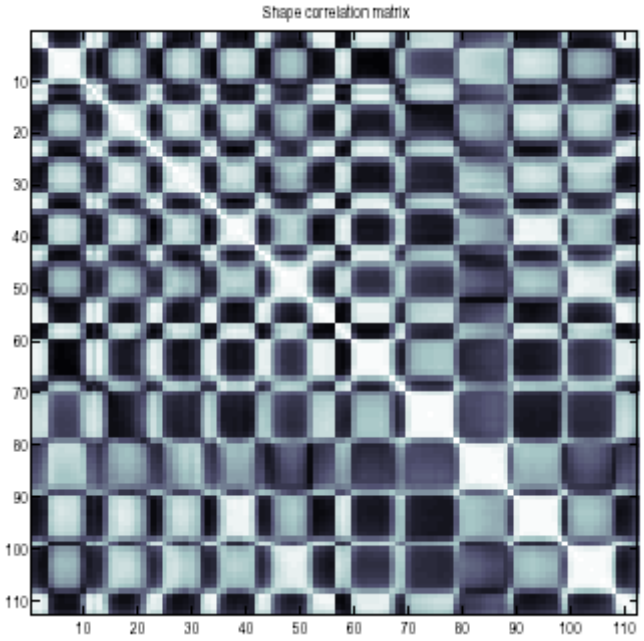


Abbildung 4.22: Korrelationsmatrix zum Hand-Beispiel, 112 Dimensionen - Graue Einträge entsprechen geringen Beziehungen, weiße oder schwarze Einträge entsprechen starken Beziehungen [SG01]

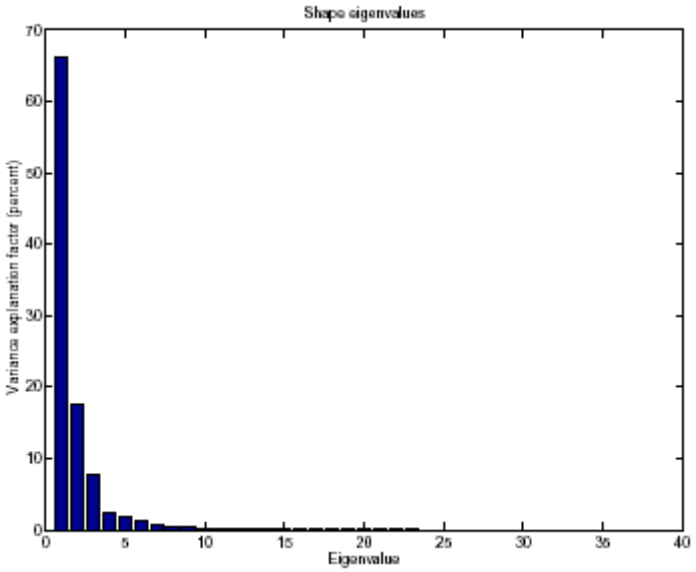
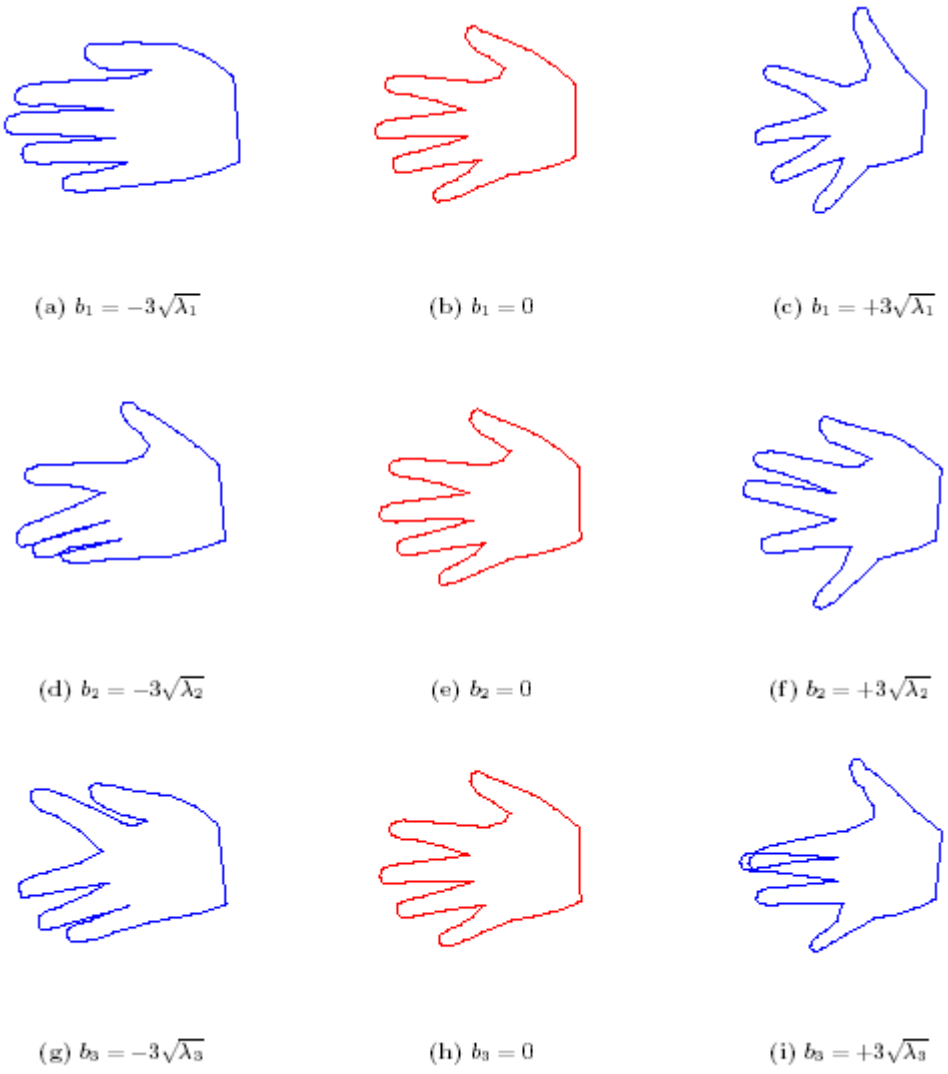


Abbildung 4.23: Einfluss der Moden [SG01]



(a) $b_1 = -3\sqrt{\lambda_1}$ (b) $b_1 = 0$ (c) $b_1 = +3\sqrt{\lambda_1}$

(d) $b_2 = -3\sqrt{\lambda_2}$ (e) $b_2 = 0$ (f) $b_2 = +3\sqrt{\lambda_2}$

(g) $b_3 = -3\sqrt{\lambda_3}$ (h) $b_3 = 0$ (i) $b_3 = +3\sqrt{\lambda_3}$

Abbildung 4.24: Darstellung der ersten drei Moden (a,b,c), (d,e,f), (g,h,i) [SG01]

lierungsfaktor.
Die Punkte $\mathbf{x}$ einer Modellinstanz im Bild lassen sich durch $\mathbf{x} = T_{X_t,Y_t,s,\theta}(\bar{\mathbf{x}} + \mathbf{\Phi b})$ mit

$$T_{X_t,Y_t,s,\theta} = \begin{pmatrix} X_t \\ Y_t \end{pmatrix} + \begin{pmatrix} s\cos\theta & s\sin\theta \\ -s\sin\theta & s\cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.7}$$

berechnen.
Zum Matching der Modellinstanz $\mathbf{x}$ mit den neuen Bildpunkten $\mathbf{Y}$ muss folgender Ausdruck minimiert werden:

$$|\mathbf{Y} - T_{X_t,Y_t,s,\theta}(\bar{\mathbf{x}} + \mathbf{\Phi b})|^2 \tag{4.8}$$

Die Fehlerquadratminimierung wird wieder durch einen iterativen Algorithmus gelöst, welcher in Algorithmus 3 dargestellt ist.

Algorithmus 3 Anpassen des Modells an neue Punkte (Matching)

1: Initialisiere $\mathbf{b}$ mit $\vec{0}$
2: Generiere Modellinstanz $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{\Phi b}$
3: Finde "beste" Parameter (X_t, Y_t, θ, s) um $\mathbf{x}$ mit $\mathbf{Y}$ zu matchen
4: Projiziere $\mathbf{Y}$ in den Form-Raum: $\mathbf{y} = T^{-1}_{X_t,Y_t,s,\theta}(\mathbf{Y})$
5: Projiziere $\mathbf{y}$ in den Tangenten-Raum von $\bar{\mathbf{x}}$ in richtiger Skalierung
6: Update der Modellparameter $\mathbf{b} = \Phi^T(\mathbf{y} - \bar{\mathbf{x}})$
7: Stelle fest, ob $\mathbf{b}$ innerhalb der Varianzen
8: Wenn noch nicht konvergiert, gehe zu 2

Der Algorithmus gliedert sich grob in drei Teile: im ersten Teil (1.-3.) wird der Mittelwert bzw. in den nachfolgenden Schritten die aktuelle Modellinstanz passend skaliert, gedreht und verschoben. Im zweiten Teil (4.-6.) wird aus $\mathbf{Y}$ ein passender Parameter $\mathbf{b}$ bestimmt, indem $\mathbf{Y}$ in den Form-Raum des Modells und den Tangenten-Raum des Mittelwerts abgebildet wird. Danach wird die Differenz zum Mittelwert festgestellt und von links mit der transponierten Eigenvektormatrix Φ multipliziert (Φ ist orthogonal!) um den Parametervektor $\mathbf{b}$ zu bestimmen. Im dritten Teil (7.-8.) wird schließlich geprüft, ob der so bestimmte Parameter $\mathbf{b}$ plausibel bzw. legal ist. Soweit das Vefahren noch nicht konvergiert, geht der Algorithmus in die nächste Iteration. Dieses Verfahren konvergiert üblicherweise innerhalb weniger Iterationen.

4.3.1.5 Active Shape Models

Im Prinzip wurden jetzt alle Vorbereitungen getroffen, um gelernte Formen in Bildern zu segmentieren. In Abschnitt 4.3.1.4 wurde beschrieben, wie man Modellinstanzen mit neuen Punkten matchen kann. Es bleibt also lediglich zu erklären, wie man an diese neuen Punkte kommt. Dazu gibt es nach [CT01a] im Wesentlichen zwei Möglichkeiten: Das Suchen an jeder Landmarke entlang der Form-Normalen nach der stärksten Kante oder nach einer bei der Modellbildung beobachteten Modellkante, wobei Letzteres in kontrastarmen Regionen wohl die bessere Wahl darstellt.
Es ist zu beachten, dass falsche (d.h. zu weit vom zu segmentierenden Objekt entfernte) Startpunkte zu falschen Ergebnissen führen können.

Algorithmus 4 Segmentierung per ASM

1: Bestimmen eines ungefähren Startpunktes im Bild
2: Auswählen eines Parameters $\mathbf{b}$
3: Kreieren einer Modellinstanz $\mathbf{X}$ im Bild
4: Suche in einer Umgebung jedes Punktes $\mathbf{X_i}$ entlang der Normalen nach einem passenden Punkt $\mathbf{X_i}'$
5: Updaten der Parameter ($\mathbf{b}, X_t, Y_t, \theta, s$) um Modellinstanz an neue Punkte anzupassen (vgl. 4.3.1.4)
6: Wenn noch nicht konvergiert, gehe zu 4

4.4 Im GenuTEP-Programm verwendete Algorithmen

Dieser Abschnitt erläutert, welche Algorithmen letztendlich zur Segmentierung der einzelnen Bildkomponenten in den C-Bogen-Aufnahmen zum Einsatz kamen, wie sie miteinander kombiniert und an die besonderen Erfordernisse des Projekts angepasst wurden.

4.4.1 Vor der Segmentierung

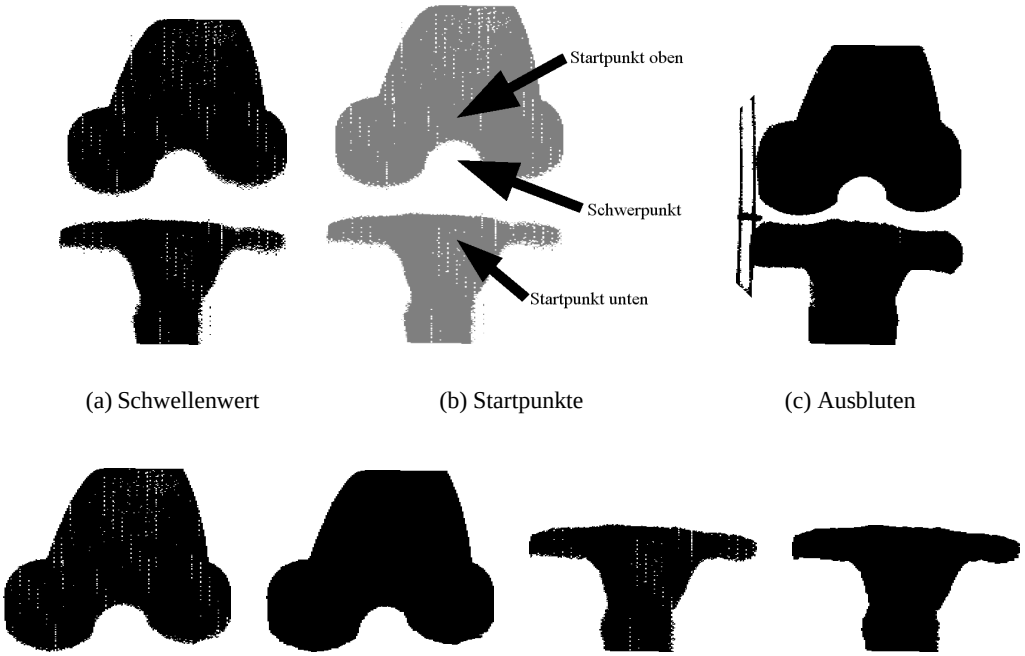
Vor der eigentlichen Segmentierung muss sichergestellt werden, dass alle Eingabebilder gewisse Rahmenbedingungen erfüllen. Eine Clippingmaske markiert den für die weitere Verarbeitung relevanten Bildbereich. Der Tonwertausgleich vermindert Helligkeitsschwankungen im Verlauf einer Bildsequenz, der Gauß-Glättungsfilter dient der Verminderung von Bildstörungen.

4.4.1.1 Clipping

Um interessante von uninteressanten Bildbereichen zu trennen und damit sicherzustellen, dass die benutzten Segmentierungsalgorithmen störungsfrei funktionieren, wird eine sog. Clippingmaske aufgebaut. Als Datenstruktur zur Aufnahme selbiger dient eine Bitmap, welche auch zum Speichern von Segmentierungsergebnissen benutzt wird. Die Bitmap besitzt das gleiche Format wie die zugrunde liegende Bildserie. Geclippte Pixel werden mit einer 1 markiert, nicht geclippte Bildelemente mit 0.
Da dem Benutzer keine neuerliche große Programminteraktion zur Definition der Clippingmaske aufgebürdet werden soll, wird zum Aufbau auf die bereits vom Benutzer verifizierten und somit absolut verlässlichen Daten über die Positionen der Punkte des Kalibrierungsgitters zurückgegriffen, welche während der C-Bogen-Kalibrierung erzeugt werden und nach Beendigung des Kalibrierungswerkzeugs (siehe Kapitel 9) in einer Datei auf der Festplatte zur Verfügung stehen. Die resultieren Daten sind C-Bogen spezifisch.
Die konvexe Hülle (siehe [B$^+$00]) der Gitterpunkte wird mithilfe der externen Bibliothek Qhull (siehe [Qhu04]) im Clippingwerkzeug bestimmt und passend in die Clippingmaske eingezeichnet. Alles innerhalb dieser Hülle ist automatisch ein für die Segmentierungsalgorithmen relevanter Bildbereich.
Der Benutzer kann der Clippingmaske manuell einen letzten projektbezogenen Feinschliff verabreichen, indem er die konvexe Hülle innerhalb der Maske Aufblasen oder Zusammenschrumpfen kann. Damit können z.B. Schatten am Bildrand beseitigt werden, welche eine Segmentierung möglicherweise negativ beeinträchtigen könnten.

4.4.1.2 Tonwertausgleich

Der Tonwertfilter dient dazu, Helligkeitsschwankungen zwischen den Bildern einer Aufnahmeserie auszugleichen. Dazu wird zunächst das Intensitätshistogramm eines Bildes berechnet. Nun wird eine Schwelle definiert, deren Wert bei $x = Bildbreite * Bildhöhe * 0.005$ liegt. x entspricht der Anzahl an sehr dunklen und sehr hellen Bildpunkten, die keine relevanten Bildinformationen enthalten. Daraufhin werden zwei Intensitätsschwellenwerte definiert, *Min* und *Max*, indem im Histogramm beginnend bei $i = 0$ die Anzahl der Bildpunkte der jeweiligen Intensitätsstufe aufsummiert wird, solange die Summe kleiner als x ist. *Min* wird dann auf das entsprechende i gesetzt. Die Vorgehensweise für *Max* ist identisch, allerdings wird von $i = 65535$ abwärts gezählt. Im Folgenden werden die dunkelsten x Bildpunkte (deren intensität kleiner als *Min* ist) auf den Wert 0 und die hellsten x Bildpunkte (intensität größer als *Max*) auf den Wert 65535 gesetzt.



(a) Schwellenwert (b) Startpunkte (c) Ausbluten

(d) Prothese oben (e) Prothese oben, geglättet (f) Prothese unten (g) Prothese unten, geglättet

Abbildung 4.25: Automatische Prothesensegmentierung

Die Bildpunkte mit Intensitäten zwischen *Min* und *Max* werden linear auf den Wertebereich 0 - 65535 abgebildet.

4.4.1.3 Gauß-Glättungsfilter

Der Gauß-Filter dient zur Glättung von Bilddaten. Er ist nötig um die Störungen der doch recht schlechten C-Bogen-Aufnahmen zu kompensieren und wird von den Segmentierungsalgorithmen für alle drei Segmente, Prothese, Stab und Diamand genutzt. Der Gauß-Filter benutzt im zweidimensionalen einen Faltungskern, der sich nach $G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$ berechnet. Es entsteht dabei eine glockenförmige Kurve, die sogenannte Gaußglocke, die im Zentrum ihr Maximum hat und zu den Rändern abfällt. Dabei kontrolliert σ die Steilheit des Abfalls. Es werden oft Gaußkern der Größe 3x3 bis 5x5 benutzt, für die GenuTEP-Applikation wurde jedoch eine variable Angabe der Kerngröße ermöglicht. Die Faltungsmaske wird dynamisch berechnet.

4.4.2 Prothese

Die (automatische) Prothesensegmentierung dient zur Vorbereitung der Stab- (siehe Kapitel 4.4.3) und Würfelsuche (siehe Kapitel 4.4.4) sowie in einer späteren Programmversion zu einer möglichen automatischen Prothesenachsenbestimmung bzw. als Eingabe für die 3D-Rekonstruktion der Prothesen mit anschließender Volumenvisualisierung.

Hier wird der Algorithmus zum automatischen Auffinden der beiden Prothesenteile erläutert. Grundannahme dabei ist, dass aufgrund der Röntgendichte der Prothesen diese im geclippten Bild die grauwertmäßig schwärzesten Inhalte darstellen. Diese Annahme dürfte auch in realen Patientenaufnahmen mit ausreichendem Kontrast gültig bleiben.
Ein erster vorbereitender Schritt besteht im Anwenden eines adaptiven Schwellenwertverfahrens auf das zu segmentierende Bild (siehe Abbildung 4.25 a). Dieses dient nur zur groben Orientierung und muss daher nicht sehr exakt arbeiten, es müssen lediglich genügend dunkle Bildelemente markiert werden. Eine ausreichende Anzahl dunkler Pixel wird derzeit aufgrund einer einfachen oberen und unteren Schranke, welche vom Algorithmus eingehalten werden soll, bestimmt, könnte aber auch anhand eines manuell modell-segmentierten Bildes festgestellt werden oder eventuell auch auf gelernten, C-Bogen-spezifischen Erfahrungswerten beruhen.
Nach dem Zwischenspeichern der Ergebnisse in einer Bitmap, wird der Schwerpunkt aller markierten Bildelemente berechnet. Dieser dient als Ausgangspunkt für ein zweifach angewandtes adaptives Bereichswachstumsverfahren in einem korrekt gedrehten Ausgangsbild (d.h. der Stab zeigt in Richtung der Nord-Süd-Achse). Die echten Startpunkte für beide Bereichswachstumsverfahren werden jeweils in einer Umgebung oberhalb bzw. unterhalb des errechneten Schwerpunkts in der Bitmap des Schwellenwertverfahrens aus Schritt eins gesucht. Dabei werden markierte Pixel gezählt und sobald eine Schranke überschritten wird startet das Bereichswachstumsverfahren (siehe Abbildung 4.25 b). Sollte es nach Anwenden bei einem der Verfahren zum Ausbluten (siehe Abbildung 4.25 c) in den anderen Teil der Prothese gekommen sein, wird der Suchbereich automatisch eingeengt und erneut gestartet. Nach Abschluss werden die beiden Ergebnisse geglättet und in Bitmaps gespeichert (siehe Abbildungen 4.25 d-g).

4.4.3 Stab

Die Stabsuche bedient sich in erster Linie der Hough-Transformation des Kantenbildes (siehe Kapitel 4.2.4.1). Der Algorithmus der Stabsuche arbeitet iterativ, zunächst in einer sehr groben Auflösung auf dem gesamten Bild, dann mit immer feineren Rasterungen auf kleineren Bildausschnitten, um eine möglichst hohe Geschwindigkeit bei gleichzeitiger Sicherstellung eines präzisen Endergebnisses zu erreichen. Algorithmus 5 skizziert zunächst den Ablauf der Stabsuche, bevor die einzelnen Schritte im Detail erläutert werden.

Algorithmus 5 Stabsuche

1: Tonwertausgleich des Bildes durch Equalizer-Filter.
2: Leichte Glättung per Gauß-Filter.
3: Berechnung eines Kantenbildes mit dem Sobel-Kantendetektor.
4: Grob quantisierte Hough-Transformation auf gesamtem Kantenbild.
5: Suche im Hough-Raum nach ausgeprägten Kanten im Abstand der Vorgabestabbreite.
6: Verfeinerung dieser Kanten durch wiederholte, immer feiner quantisierte Hough-Transformation im immer weiter eingeschränkten Bildausschnitt.
7: Suche nach der Skalierungsmarke durch einfachen Intensitätsvergleich ausgewählter Pixel in der Umgebung des Stabes.

Der Algorithmus kann a priori-Wissen auswerten, um stabiler zu laufen. Arbeitet er auf einer Serie von Bildern, so benutzt er Informationen aus dem ersten automatisch segmentierten oder manuell vorgegebenen Bild über die Stabbreite und die Intensität der Stabkanten im Kantenbild, um seinen Suchraum besser einschränken zu können.

Im Folgenden werden die einzelnen Schritte der Stabsuche näher erläutert.

1. Auf jedes zu segmentierende Bild wird zunächst ein Tonwertausgleich angewandt (siehe Kapitel 4.4.1.2), um die durch die dynamische Anpassung der Strahlungsintensität während der Aufnahme der Bilder entstandenen Intensitätsschwankungen auszugleichen.

2. Dann wird das Bild mittels Gauß-Filter (siehe Kapitel 4.4.1.3) leicht geglättet, um kleinere Bildstörungen herauszufiltern.

3. Schließlich wird per Sobel-Filter (siehe Kapitel 4.2.2.3) ein Kantenbild erzeugt, mit dem im Folgenden weitergearbeitet wird.

4. Nun wird zunächst eine globale Linien-Hough-Transformation durchgeführt. Dabei werden nur die Pixel des Kantenbildes als gesetzt interpretiert, deren Intensität in einem vorher festgelegten Grenzbereich liegt. Der voreingestellte Bereich verläuft von 25000 bis 60000 (im 16 Bit-Graubild, mögliche Werte liegen also zwischen 0 und 65535), wird aber nach erfolgreicher Segmentierung des ersten Bildes einer Serie angepasst. Somit werden alle Kanten mittlerer Stärke zuverlässig erkannt, extrem starke Kanten, wie sie insbesondere bei der Prothese auftreten, werden ignoriert. Diese erste Transformation sucht Geraden, deren Normalenvektor zwischen -25° und 25° liegt, bei einer Schrittweite von 0.5°. Um diese globale Transformation etwas zu beschleunigen, wird zunächst nur jede zehnte Zeile des Ausgangsbildes untersucht. Das Ergebnis ist für eine erste Näherung ausreichend.

5. Nun wird eine sortierte Liste mit bis zu 50 Maxima des Hough-Raums erstellt. Diese wird, beim größten Wert beginnend, so lange durchlaufen, bis zu einem Wert (der ja eine Gerade im Bild darstellt) ein korrespondierender Wert im Hough-Raum gefunden wird, mit dem der aktuelle Wert einen Stab plausibler Breite bilden kann. Die Vorgabestabbreite wird aus dem ersten erfolgreich segmentierten Bild einer Serie bezogen, sonst wird ein Defaultwert abhängig von der Bildgröße berechnet.

6. Wurden im ersten Schritt zwei potentielle Stabkanten gefunden, wird das Ergebnis nun verfeinert. Ab jetzt wird der Suchraum in jedem Schritt weiter eingeschränkt. Der Winkelbereich wird auf das aktuelle Zwischenergebnis +/- 1.5 $\times$ die aktuelle Winkelauflösung eingestellt. Das entspricht im Hough-Raum der Spalte, in der das aktuelle Maximum gefunden wurde, zzgl. beider daneben liegender Spalten. Die Winkelauflösung wird halbiert, so dass die neue Hough-Tabelle aus 6 Spalten besteht. Es wird nicht mehr das komplette Bild analysiert, sondern nur noch der Bereich, in dem sich die verfeinerte Kante befinden kann. Dazu werden aus der aktuellen Länge des Normalenvektors, sowie den beiden neuen Grenzen des Winkelbereichs neue Start- und Endwerte für den zu untersuchenden Bereich der x-Achse ermittelt. Es wird also in jedem Verfeinerungsschritt ein immer schmaler werdender vertikaler Ausschnitt des Bildes nach Geraden mit einem Normalenvektor eines immer kleiner werdenden Winkelbereichs untersucht, bis das Ergebnis konvergiert. Wurden beide Stabkanten verfeinert wird noch überprüft, ob sie jetzt aufeinander liegen. Das kann bei qualitativ schlechten Bildern passieren, in denen die eigentlichen Stabkanten nicht sehr stark ausgeprägt sind. Dort könnten im ersten Schritt zwei „falsche“ Kanten gefunden werden, die so schief im Bild liegen, dass bei der Verfeinerung noch beide Kanten im verkleinerten Bildbereich liegen. Da bei den Verfeinerungsschritten stets nur die stärkste gefundene Kante weiter verfolgt wird, würden beide im ersten Schritt gefundenen Kanten

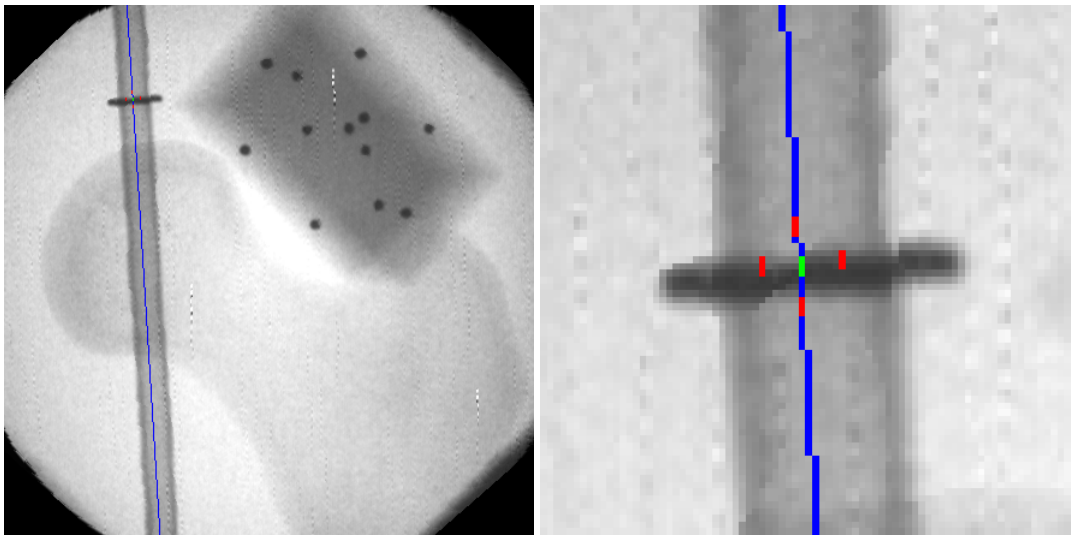


Abbildung 4.26: Stabsuche: Skalierungsmarke. Visualisierung der Punkte, die zur Erkennung der Skalierungsmarke überprüft werden.

nach dem zweiten Schritt zusammenfallen. Sollten die gefundenen Kanten nach Abschluss dieses Schrittes identisch sein, bricht der Algorithmus ab und das Ergebnis wird verworfen.

7. Wurden zwei Stabkanten gefunden, muss noch die genaue Lage der Skalierungsmarke bestimmt werden. Dazu wird die Stabmitte von oben nach unten durchlaufen (entlang der blauen Linie in Abbildung 4.26) und nach Minima im ursprünglichen Bild (nicht im Kantenbild!) durchsucht. Wird ein neues Minimum gefunden, das kleiner als alle evtl. bereits gefundenen ist, wird geprüft, ob es sich um den Ring handelt, der die Skalierungsmarke darstellt. Zu diesem Zweck wird überprüft, ob zwei Pixel links und rechts des Minimums ebenfalls kleine Werte aufweisen, während zwei Pixel ober und unterhalb des Minimums hohe Intensitätswerte aufweisen müssen (diese vier Umgebungspixel sind im Bild rot dargestellt). Auf diese Art und Weise lässt sich die Skalierungsmarke ganz gut charaktierisieren. Die eigentliche Position der gefundenen Marke ist im Bild grün dargestellt. Insgesamt wurden drei mögliche Marken gefunden, die unterste stellt dabei das Endergebnis dar.

4.4.4 Diamant

Die Segmentierung des Kalibrierobjektes geschieht in zwei einzelnen Schritten. Zunächst werden die Punkte des Objekts gesucht, dann werden die 12 Punkte in einem zweitem Schritt korrekt numeriert, damit die 3D Rekonstruktion korrekte Eingaben erhält. Dieser zweite Schritt ist auch nach manueller Segmentierung der Punkte anwendbar.

4.4.4.1 Automatische Punktsuche

Die Punkte des Kalibrierobjektes werden mit Hilfe einer Kreis-Hough-Transformation (Kapitel 4.2.4.2) gefunden. Zur korrekten Funktion muss die „Clipping Maske“ (Kapitel 4.4.1.1) für diesen Algorithmus den Randbereich der Bilder, den Stab sowie die Prothesen ausmaskieren,

anderenfalls sind die Ergebnisse falsch. Da die Punkte auf den vorhandenen Aufnahmen etwa genauso groß sind, wie der zur Filterung benötigte Gaußkegel, erzeugen konstrastreiche Kanten in etwa die gleiche „Antwort“ im Houghraum, wie die gesuchten Punkte.
Der Algorithmus trägt nur nicht geclippte Punkte in den Houghraum ein und durchsucht den Raum nach dem Maximum. Ein adaptiv und mehrschrittig arbeitender Suchalgorihmus findet zunächst alle Werte im Houghraum, die größer als eine Schwelle sind und an deren Stelle bislang noch keine Kreise gefunden worden sind. Diese werden in einer Lineare Liste gespeichert und in einem zweiten Schritt werden „Nachbarn“ im Umkreis von 3 Pixeln zusammengefasst, d.h. der Schwerpunkt aller Punkte wird in die endgültige Liste eingetragen.
Dieses Vorgehen wird nun solange ausgeführt und die Schwelle verringert oder Punkte gelöscht, bis entweder eine interne Abbruchschranke (mehr als 128 Iterationen) erreicht wurde, oder zwölf Punkte gefunden worden sind.
Die Punkte werden auch dann ausgegeben, falls nicht alle zwölf Punkte gefunden worden sind, da oftmals fehlende Punkte leicht manuell hinzugefügt werden können.

4.4.4.2 Automatische Punktzuordnung

Die Diamantpunktzuordnung folgt keinem bekanntem Segmentierungsalgorithmus, sondern wurde aufgrund von Beobachtungen und Modellannahmen für das Programm entwickelt. Der Ablauf erfolgt in mehreren Schritten, die teilweise auch scheitern können, sei es wegen falsch segmentierten Punkten oder aufgrund einer Unlösbarkeit der Situation. In diesen Fällen wird eine Exception mit einem passendem Text für den Benutzer ausgelöst, die ihm vom Programm angezeigt wird.

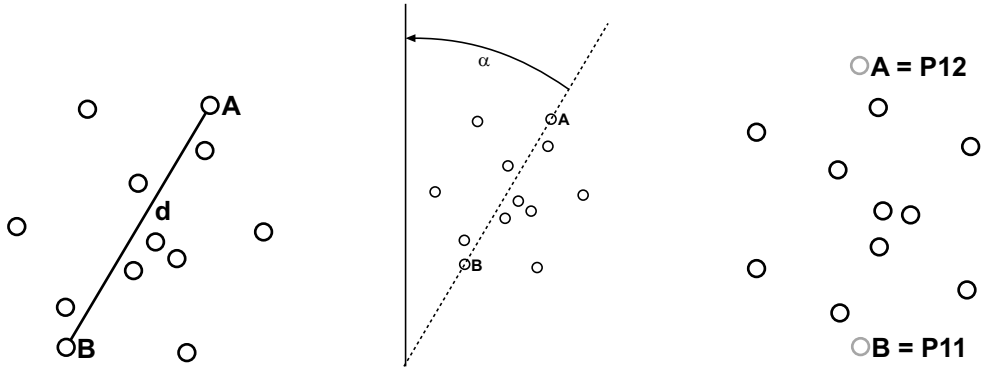
Im ersten Schritt (siehe Algorithmus 6) werden die beiden entferntesten Punkte als die Spitzpunkte des Diamanten angenommen. Wenn diese beiden Punkte nicht die Spitzpunkte sind, ist das Ergebnis der Zuordnung auf jeden Fall falsch. Dies kann zum Beispiel auftreten, wenn die 2D-Projektion nicht linear ist, d.h. die Bilder nicht korrekt entzerrt wurden.

Algorithmus 6 Diamantpunktzuordnung Schritt 1: Ausrichten

Benötigt: $\mathcal{P} := \left\{\mathbf{p}_i \in I\!R^2 \mid i \in \{1 \dots 12\}\right\}$: Eingabemenge der Würfelpunkte
Benötigt: $d(\mathbf{p}_i, \mathbf{p}_j)$: euklidisches Distanzmaß
1: finde Punktepaar $(\mathbf{p}_A, \mathbf{p}_B) := \arg\max_{i,j} \left(d(\mathbf{p}_i, \mathbf{p}_j)\right)$ (Abb. 4.27(a))
2: ermittle $\alpha := \angle([0\ 1]^T, \mathbf{p_B} - \mathbf{p_A})$
3: rotiere die gegebene Punktemenge um den Winkel α (Abb. 4.27(b))
4: (O.B.d.A.) setze P11 auf $\mathbf{p}_{11} := \arg\min\left((\mathbf{p}_A)_y, (\mathbf{p}_B)_y\right)$
5: setze P12 auf $\mathbf{p}_{12} := \arg\max\left((\mathbf{p}_A)_y, (\mathbf{p}_B)_y\right)$ (Abb. 4.27(c))
6: setze $\mathcal{P}_2 = \mathcal{P} \setminus \{P11, P12\}$
7: Kantenzuordnung($\mathcal{P}_2$) (siehe Algorithmus 7)

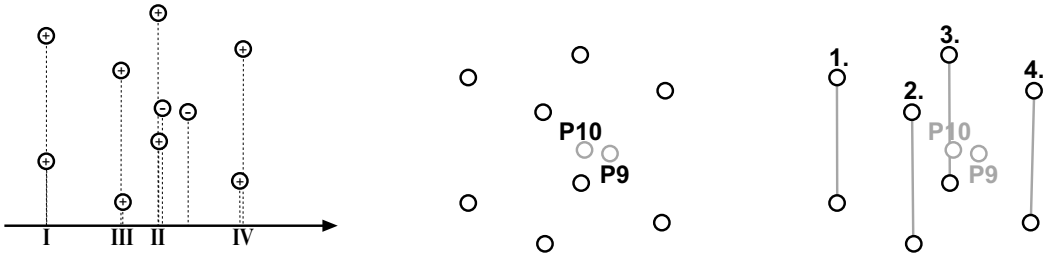
In Schritt zwei (siehe Algorithmus 7) werden die Punkte aufgrund ihrer Nachbarschaft entsprechenden Kanten zugeordnet. Dabei können folgende Fehler auftreten:

- es werden mehr als zwei Punkte einer Kante zugeordnet:
    1. Ein (oder zwei) „innerer(e)“ Punkt(e) liegen im Bild auf einer „echten“ Kante.



(a) Die entferntesten Punkte (b) Ausrichten (c) Punkt 11 und 12

Abbildung 4.27: Diamantpunktzuordnung Schritt 1: Ausrichten



(a) Projektion auf die X-Achse (b) Die beiden inneren Punkte 9 und 10 (c) Kantenzuordnung

Abbildung 4.28: Diamantpunktzuordnung Schritt 2: Kantenzuordnung

2. Zwei Kanten sind Deckungsgleich.
3. Das Bild ist so verzerrt, das eine (oder mehrere) Kanten nicht korrekt gefunden werden.

- es werden keine vier Kanten gefunden.

in diesen Fällen bricht der Algorithmus mit einem Fehler ab. Die beiden nichtzugeordneten Punkte werden als „innere“ Punkte angenommen, wobei der Punkt mit der kleineren Y-Koordinate als Punkt 10 angenommen wird (Abbildung 4.28(b)). Sollte dies falsch sein, wird der Diamant im nächsten Schritt falsch zugeordnet.
Der letzte Schritt (siehe Algorithmus 8) basiert auf einer Quadranteneinteilung (siehe Abbildung 4.29(c)), deren allgemeine Korrektheit noch zu beweisen wäre. Dabei wird die Lagebeziehung zwischen den inneren Punkten und dem Mittelpunkt ausgewertet (Abbildung 4.29(b)). Dadurch können die beiden äußeren Kanten nummeriert werden (Abbildung 4.29(d)), durch einen Längenvergleich der beiden mittleren Kanten ist derenen Beziehung im Raum („vorne“ vs. „hinten“) bekannt und auch diese können nummeriert werden (Abbildung 4.29(e)). Es sei hier auf die

Algorithmus 7 Diamantpunktzuordnung Schritt 2: Kantenzuordnung

Benötigt: $\mathcal{P} := \{\mathbf{p}_i \in I\!R^2 \mid i \in \{1 \ldots 10\}\}$: Eingabe(teil)menge der Würfelpunkte
Benötigt: $d(\mathbf{p}_i, \mathbf{p}_j)$: euklidisches Distanzmaß

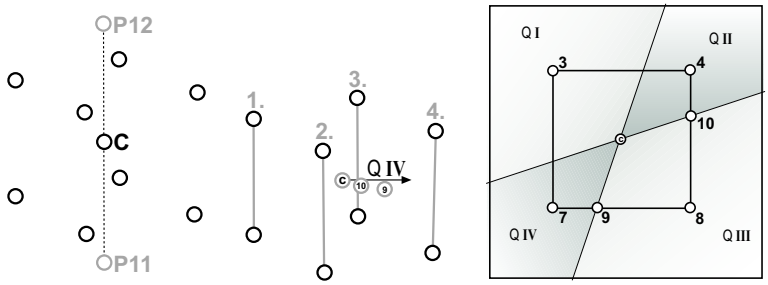
1: $\forall \mathbf{p}_i \in \mathcal{P} : (\mathbf{p}_i)_y := 0$ (Abb. 4.28(a))
2: markiere alle $\mathbf{p}_i \in \mathcal{P}$ als *nicht zugeordnet*
3: `cluster` := 0
4: `zugeordnet` := 0
5: `minX` := 0
6: `maxX` := $2/3 \cdot \max_{i,j} d(\mathbf{p}_i, \mathbf{p}_j)$
7: **while** (`zugeordnet` < 8) und (`minX` < `maxX`) und (`cluster` < 4) **do**
8: **for** $\mathbf{p}_i \in \mathcal{P}$ die *nicht zugeordnet* sind **do**
9: **for** $\mathbf{p}_j \in \mathcal{P}$ mit $j \neq i$ **do**
10: **if** $d(\mathbf{p}_i, \mathbf{p}_j) \leq$`minX` **then**
11: **if** $\mathbf{p}_j$ ist *zugeordnet* **then**
12: füge $\mathbf{p}_i$ in Cluster von $\mathbf{p}_j$ ein und markiere $\mathbf{p}_i$ als *zugeordnet*
13: `zugeordnet` := `zugeordnet`+1
14: **else**
15: bilde Cluster aus $\mathbf{p}_i$ und $\mathbf{p}_j$ und markiere beide Punkte als *zugeordnet*
16: `zugeordnet` := `zugeordnet`+2
17: `cluster` := `cluster`+1
18: **end if**
19: **end if**
20: **end for**
21: **end for**
22: `minX` := `minX+1`
23: **end while**
24: {Die Ausgabe des Algorithmus besteht im Erfolgsfalle aus 4 Clustern, welche jeweils 2 Punkte enthalten die eine Kante bilden (Abbildung 4.28(c)), siehe dazu auch Erläuterungen im Text}
25: Nummerierung($\mathcal{P}, \mathcal{K}$) (siehe Algorithmus 8)

zunächst verwirrende Eigenschaft von Röntgenbildern hingewiesen, dass näher am Detektor liegende Kanten kürzer sind, als weiter entfernte (Abbildung 4.30). Unter Annahme der Korrektheit des Quadrantenmodells kann im letzten Schritt nur durch den Längenvergleich ein Fehler auftreten, z.B. wenn die Bilder verzerrt sind, dies ist aber durch ein geeignetes Dewarping (Vgl. Kapitel 3) auszuschließen.

Algorithmus 8 Diamantpunktzuordnung Schritt 3: Nummerierung

Benötigt: $\mathcal{P} := \{\mathbf{p}_i \in I\!R^2 \mid i \in \{1 \dots 12\}\}$: Eingabemenge der Würfelpunkte
Benötigt: $\mathcal{K} := \{\mathbf{k}_i \in I\!N \mid i \in \{1 \dots 12\}\}$: Kantenzuordnung der Punkte
Benötigt: $d(\mathbf{p}_i, \mathbf{p}_j)$: euklidisches Distanzmaß
1: $\forall i \in \{1 \dots 8\}$: Sortiere Punkte $\mathbf{p}_i$ nach aufsteigender X-Koordinate
2: $\mathcal{L} := \{\forall i, j : \mathbf{l}_{\mathbf{k}_i} = d((\mathbf{p}_{\mathbf{k}_i}, \mathbf{p}_{\mathbf{k}_j}) \mid \mathbf{k}_i = \mathbf{k}_j\}$: Errechne Kantenlängen
3: $\mathbf{c} = 1/2 * (\mathbf{p}_{11} - \mathbf{p}_{12})$ (Abb. 4.29(a))
4: **if** $((\mathbf{p}_{10})_x < \mathbf{c}_x)$ **then**
5: **if** $((\mathbf{p}_9)_x < \mathbf{c}_x)$ **then**
6: Quadrant II: P8 links, P3 rechts, P4 vorne, P7 hinten
7: **else**
8: Quadrant I: P4 links, P7 rechts, P3 vorne, P8 hinten
9: **end if**
10: **else**
11: **if** $((\mathbf{p}_9)_x < \mathbf{c}_x)$ **then**
12: Quadrant III: P7 links, P4 rechts, P8 vorne, P3 hinten
13: **else**
14: Quadrant IV: P3 links, P8 rechts, P7 vorne, P4 hinten (Abb. 4.29(e))
15: **end if**
16: **end if**

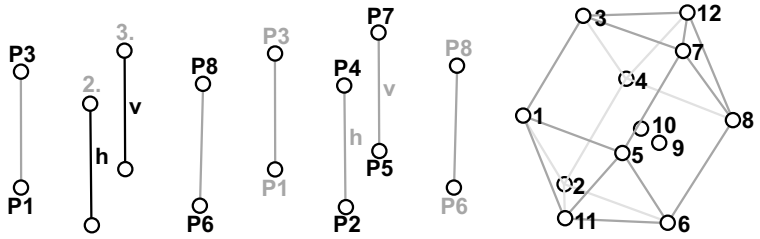
Beim Testen der Anwendung hat sich dieser Algorithmus als stabil erwiesen. Alle beobachteten und nachvollzogenen Fälle in denen die Punkte falsch zugeordnet wurden, waren auf falsche oder fehlerhafte Vorverarbeitung der Bilder zurückzuführen. In der Praxis ist dies vor allem auf eine starke Verzerrung im Randbereich zurückzuführen, ein verbessertes Dewarping wäre wünschenswert, dies ist allerdings aufgrund fehlender Stützstellen außerhalb des Bildbereichs nicht trivial. Abbildung 4.29(e) zeigt den Diamanten, wie er von dem Algorithmus zugeordnet wird, das Bild ist einer aktuellen Bildsequenz entnommen.



(a) Mittenbestimmung

(b) Quadrantenzuordnung

(c) Quadrantenherleitung (Aufsicht)



(d) Nummerierung der beiden äußeren Kanten

(e) Nummerierung der inneren Kanten nach Längenvergleich

(f) Vollständig zugeordnete Punkte

Abbildung 4.29: Diamantpunktzuordnung Schritt 3: Nummerierung

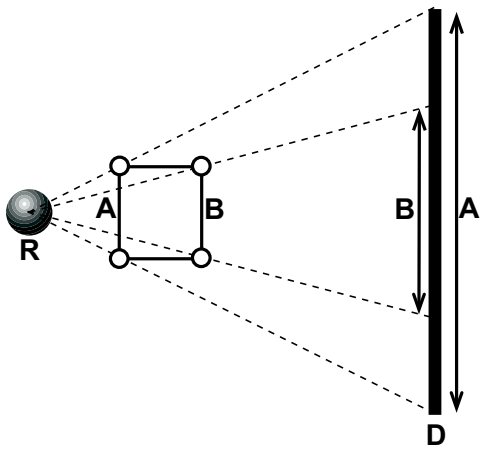


Abbildung 4.30: Längenverhältnisse in Röntgenbildern. Eine Röntgenquelle (R) sendet Strahlen aus, die auf den Diamanten treffen. Die Detektorfläche (D) erfasst die eintreffende Strahlung und liefert das Bild. Im Gegensatz zu einer optischen Kamera ist beim Röntgenverfahren Eine entfernte Kante (A) ist im Bild länger als eine näher am Detektor gelegene Kante (B).

Kapitel 5

3D Rekonstruktion

5.1 Ziele

Das Ziel dieser Programmkomponente besteht darin, anhand eingegebener zweidimensionaler Daten eine vollständige dreidimensionale geometrische Beschreibung der Bein- und Prothesenachsen zu berechnen. Diese beinhaltet insbesondere die genauen relativen Positionen der Ober- und Unterschenkel- sowie der Prothesenachsen. Anschließend lässt sich mit Hilfe dieses Rekonstruktionsmodells die Abweichung der Prothesen- von den Schenkelachsen berechnen.

5.2 Vorgehensweise

Die Definition nach [H$^+$99] dient als Grundlage für die Bestimmung der Schenkelachsen, welche sieben anatomische Punkte zur Bestimmung der Achsen verwendet. Im Hüftbereich ist dies der Femurkopf, im Knie dienen laterale und mediale Epikondyle, Femurköpfchen und Tuberositas Tibiae als Referenz sowie lateraler und medialer Malleolus im Sprunggelenkbereich. Für die Rekonstruktion der Prothesenachsen sind herstellerspezifische Angaben notwendig.
Aufgrund des relativ eingeschränkten Aufnahmebereichs handelsüblicher C-Bögen müssen – um alle Punkte erfassen zu können – Aufnahmen der drei erwähnten anatomischen Bereiche (Hüfte, Knie und Knöchel) angefertigt werden, welche zunächst unabhängig voneinander rekonstruiert werden. Da allerdings aus einem einzelnen Bild keine dreidimensionalen Informationen gewonnen werden können, müssen in jedem anatomischen Abschnitt Bilder aus mindestens drei verschiedenen Perspektiven aufgenommen werden. Weiterhin ist es für eine dreidimensionale Rekonstruktion zweidimensionaler Fluoroskopieaufnahmen erforderlich, bei der Aufnahme der Bilder Kalibrierobjekte einzubeziehen. Im Rahmen des Projektes werden drei verschiedene Kalibriergegenstände verwendet, welche eigens zu diesem Zweck angefertigt wurden (siehe Abschnitt 2.1.2).
Mit Hilfe eines Kalibrierungsgitters werden im ersten Programmabschnitt die intrinsischen Kameraparameter ermittelt (vergleiche Kapitel 3). Diese bestimmen u.a. die Brennweite und sind bei der Verwendung eines C-Bogens für alle Bilder gleich. Der zweite Kalibriergegenstand ist ein Quader aus Acryl, in welchem an genau definierten Stellen zwölf kleine Metallkugeln diamantenförmig positioniert sind. Dieser Diamant wird bei der Anfertigung der Fluoroskopieaufnahmen neben dem Patienten auf der Tischplatte platziert, so dass er in allen Aufnahmen vollständig sichtbar ist. Anhand der Lage der aufgenommenen Kalibrierungskugeln kann mit Hilfe des Rekonstruktionsalgorithmus' auf die tatsächliche dreidimensionale Lage der Szene geschlossen

werden.
Um die drei so rekonstruierten anatomischen Bereiche zueinander in Beziehung zu setzen, ist es erforderlich, dass ein- und dasselbe Objekt in allen drei Aufnahmesequenzen sichtbar ist. Diese Aufgabe erfüllt der Kalibrierungsstab, welcher ebenfalls neben das Bein des Patienten auf den Tisch gelegt wird.
Nachdem die drei rekonstruierten Einzelbereiche aneinander ausgerichtet und somit zu einer einzigen dreidimensionalen Beschreibung des Beines zusammengefügt worden sind, kann der Winkel zwischen Prothesen- und Schenkelachsen berechnet werden.
Bevor allerdings auf die Einzeleinheiten des Verfahrens eingegangen wird, werden zunächst einige mathematische Grundlagen erläutert.

5.3 Grundlagen

5.3.1 Projektiver Raum

Im projektiven Raum werden *n*-dimensionale Punkte durch *n+1* sogenannte homogene oder projektive Koordinaten dargestellt. Die Punkte sind dabei lediglich bis auf einen skalaren Faktor bestimmt, d.h. zwei Punkte sind dann gleich, wenn es ein $\lambda \neq 0$ gibt, so dass gilt: $\mathbf{x} = \lambda\mathbf{y}$. Der Zusammenhang zwischen euklidischen Koordinaten $(y_1, y_2, \ldots, y_n)^T$ eines Punktes und dessen Koordinaten im projektiven Raum $(x_1, x_2, \ldots, x_{n+1})^T$ lässt sich wie folgt darstellen:

$$(x_1, x_2, \ldots, x_{n+1})^T = \left(\frac{x_1}{x_{n+1}}, \ldots, \frac{x_n}{x_{n+1}}, 1\right)^T \leftrightarrow (y_1, \ldots, y_n)^T .$$

Die Vorteile der Verwendung projektiver Koordinaten bestehen darin, dass auch Translationen als Matrizenprodukt formulierbar sind und einige Rechnungen durch die Dualität von Punkt und Gerade vereinfacht werden können.

5.3.2 Das Lochkamera-Modell

Den folgenden Überlegungen liegt das Lochkameramodell (engl. *pinhole model*) zugrunde, welches nun kurz erläutert wird. Eine schematische Darstellung findet sich in Abbildung 5.1. Das Modell besteht aus zwei parallelen Ebenen, der Bildebene *I* und der Blende. Das Loch in der Blende wird dabei als Projektionszentrum *C* (engl. *optical center*) der Kamera bezeichnet; der Abstand zwischen *C* und der Bildebene *I* heißt Brennweite *f* (engl. *focal length*). Die durch den Punkt C und senkrecht zur Bildebene verlaufende Gerade wird optische Achse genannt. In Zukunft wird die Bildebene vor der Blende eingezeichnet, da das Bild so nicht um 180° gedreht abgebildet wird und es für die Geometrie keine Rolle spielt.

5.3.3 Abbildung von Welt- in Pixelkoordinaten

Die Abbildung von Welt- in Pixelkoordinaten ist gegeben durch:

$$\tilde{\mathbf{m}} = \mathbf{APD}\tilde{\mathbf{M}}_W .$$

Im Einzelnen bedeutet das:

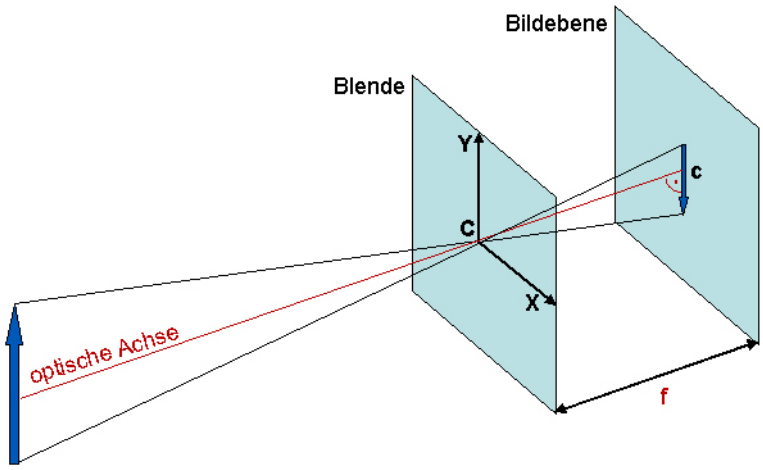


Abbildung 5.1: Lochkameramodell

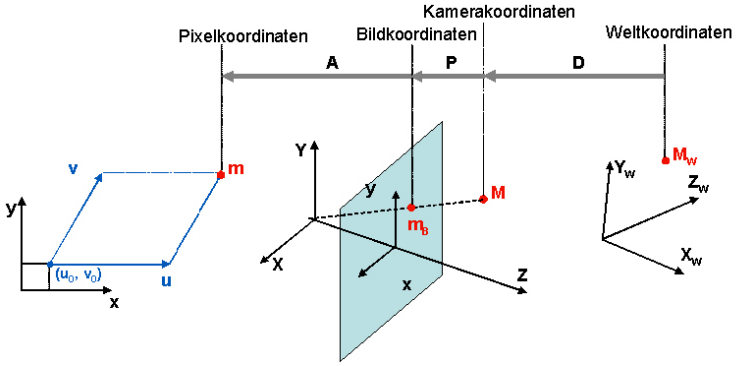


Abbildung 5.2: Abbildung von Welt- in Pixelkoordinaten

1. Der gegebene Punkt $\mathbf{M}_W$ wird - mittels Matrix $\mathbf{D}$ - vom Weltkoordinatensystem in das Kamerakoordinatensystem $(C; X, Y, Z)$ transformiert. Weltkoordinaten werden im Folgenden mit Index “W“ gekennzeichnet. Die dazu benötigte Transformation $\mathbf{D}$ enthält die extrinsischen Parameter der Kamera.

2. Der Punkt wird aus dem Kamerakoordinatensystem ins Bildkoordinatensystem (Index “B“) projiziert. Dies geschieht mit der Projektionsmatrix **P**.

3. Mit Hilfe der internen Kameraparameter (Matrix **A**) erfolgt eine Umrechnung in Pixelkoordinaten $((u_0, v_0); u, v)$.

Extrinsische Parameter

Die extrinsischen Parameter (engl. *extrinsic parameters*) einer Kamera spezifizieren die Lage der Kamera im Weltkoordinatensystem. Um einen Punkt $\tilde{M}_W$ im Weltkoordinatensystem in die entsprechende Darstellung $\tilde{M}$ bezüglich des Kamerakoordinatensystems zu überführen, ist eine Multiplikation mit der regulären 4×4-Matrix **D** erforderlich. Diese setzt sich im Allgemeinen

aus einer Starrkörperrotation und einer Translation zusammen (euklidische Transformation):

$$\mathbf{D} = \begin{pmatrix} \mathbf{R}_{3\times 3} & \mathbf{t}_3 \\ \mathbf{0}_3^T & 1 \end{pmatrix}$$

Die drei in der Rotationsmatrix $\mathbf{R}_{3\times 3}$ enthaltenen Drehwinkel um die jeweiligen Raumachsen sowie die drei Translationsparameter aus $\mathbf{t}_3$ werden als extrinsische Parameter der Kamera bezeichnet.

Perspektivische Projektion

Die perspektivische Projektion bildet einen Punkt aus dem dreidimensionalen Kamerakoordinatensystem in das zweidimensionale Bildkoordinatensystem ab. Dabei ist das optische Projektionszentrum C der Ursprung des Kamerakoordinatensystems; die Z-Achse entspricht der optischen Achse (siehe Abbildung 5.1). Der Ursprung des Bildkoordinatensystems c befindet sich im sogenannten Bildhauptpunkt, dem Schnittpunkt von Bildebene und optischer Achse. Die x- und y-Achsen sind parallel zu den X- und Y-Achsen im Kamerakoordinatensystem.
Die Beziehung zwischen den beiden Koordinatensystemen ergibt sich wie folgt aus dem Strahlensatz:

$$\frac{x}{X} = \frac{y}{Y} = \frac{f}{Z}.$$

Durch Umformung erhält man für die Bildkoordinaten

$$x = \frac{f}{Z}X \quad \text{und} \quad y = \frac{f}{Z}Y.$$

Dies kann – ausgedrückt in homogenen Koordinaten – mit folgender Gleichung dargestellt werden:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \underbrace{\begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\mathbf{P}} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}.$$

Dabei stellt **P** die perspektivische Projektionsmatrix dar.

Die Beziehung zwischen den beiden Koordinatensystemen kann nun insgesamt mit folgender Gleichung beschrieben werden:

$$s\tilde{\mathbf{m}} = \mathbf{P}\tilde{\mathbf{M}}. \tag{5.1}$$

s ist ein beliebiger Skalar ungleich 0, welcher verdeutlicht, dass die verwendeten homogenen Koordinaten nur bis auf einen skalaren Faktor definiert sind.

Intrinsische Parameter

Die Matrix

$$\mathbf{A} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.2}$$

enthält die intrinsischen Parameter der Kamera und stellt eine Transformation von Bildkoordinaten in Pixelkoordinaten gemäß

$$\tilde{\mathbf{m}} = \mathbf{A}\tilde{\mathbf{m}_B}$$

dar. Dabei geben u_0 und v_0 die Koordinaten des Bildhauptpunktes c an. α und β sind Skalierungsfaktoren, γ die Scherung der beiden Achsen.
Die fünf intrinsischen Parameter sind komplett unabhängig von der Position und Orientierung der Kamera.

Insgesamt ergibt sich als Abbildung von Welt- in Pixelkoordinaten:

$$\mathbf{APD} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{R}_{3\times 3} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}. \tag{5.3}$$

5.4 Rekonstruktion mit affiner Iteration bzw. linearer Approximation

Dieser Rekonstruktionsalgorithmus liefert einerseits die Positionen der Kugeln des Kalibrierquaders als dreidimensionale Spaltenvektoren. Die Matrix, die sich aus diesen Vektoren zusammensetzt, wird im übrigen in den Ausarbeitungen zu diesem Thema [CH94] als Form **F** bezeichnet. Andererseits berechnet der Algorithmus die Kamerapositionen, indem die extrinsischen Matrizen jeder Kamera rekonstruiert werden. Eine extrinsische Matrix berechnet sich aus einer 2×3-Matrix, an deren Berechnung bestimmte Bedingungen geknüpft werden. Diese Matrix wird als Bewegungsmatrix **B** bezeichnet. Der Grund für die Bezeichnung „Bewegungsmatrix“ ist der, dass die extrinsische Matrix an der Form eine Rotation und eine Verschiebung im Weltkoordinatensystem durchführt. Da die extrinsische Matrix ansonsten keinerlei Skalierung oder Scherung an der Form verursacht, wird diese Art der Transformation als euklidisch bezeichnet (Transformation starrer Körper durch Rotation und Translation ohne Veränderung der Form durch Skalierung oder Scherung).

Für die beiden ersten Bildkoordinaten x_i und y_i des Punktes $\tilde{\mathbf{m}}_B$ wird die Abbildung $\mathbf{P} \cdot \mathbf{D}$ von Welt- in Bildkoordinaten (siehe Gleichung 5.3) in geschlossene Formeln gebracht. Dadurch können die beiden resultierenden Formeln so ausgedrückt werden, dass der Einfluss der Projektion als linearer Faktor $1/(1+\epsilon_i)$ in die Abbildung eingeht. Dieser Faktor ist seinerseits ein Term, der nichtlinear von der z-Koordinate von $\tilde{\mathbf{m}}_B$ abhängt:

$$\mathbf{m}_B = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \mathbf{I} \cdot \mathbf{M}_W + x_0 \\ \mathbf{J} \cdot \mathbf{M}_W + y_0 \end{pmatrix} \frac{1}{1+\epsilon} \tag{5.4}$$

mit

$$\begin{aligned} \mathbf{I} &= \mathbf{i}^T/t_z \\ \mathbf{J} &= \mathbf{j}^T/t_z \\ \epsilon &= \mathbf{k} \cdot \mathbf{M}_W/t_z \end{aligned} \qquad \begin{aligned} x_0 &= t_x/t_z \\ y_0 &= t_y/t_z \end{aligned} \qquad \begin{pmatrix} \mathbf{i}^T & t_x \\ \mathbf{j}^T & t_y \\ \mathbf{k}^T & t_z \\ \mathbf{0}_3^T & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R}_{3\times 3} & \mathbf{t}_3 \\ \mathbf{0}_3^T & 1 \end{pmatrix} \tag{5.5}$$

Der zweidimensionale Vektor $\mathbf{m}_B$ entsteht durch homogene Division durch die z-Komponente aus dem Vektor $\tilde{\mathbf{m}}_B$. Ebenso entsteht der dreidimensionale Vektor $\mathbf{M}_W$ durch homogene Division aus $\tilde{\mathbf{M}}_W$. Die beiden dreidimensionalen Vektoren **I** und **J** bilden im übrigen die Zeilen der euklidisch rekonstruierten (2×3)-Bewegungsmatrix $\mathbf{B}_{rigid}$.

Der in diesem Programm verwendete Algorithmus approximiert beide Lösungen (Bewegungsmatrix, Form) einer dreidimensionalen, euklidischen Rekonstruktion simultan. Um die Aufgabe zu lösen kann ein perspektivisches Kameramodell oder dessen affine Approximation (paraperspective) verwendet werden:

$$\underbrace{\frac{1}{1+\epsilon}}_{\text{perspektivisch}} \quad \approx \quad \underbrace{1-\epsilon}_{\text{para-perspektivisch}} \tag{5.6}$$

Das bedeutet, dass sich der para-perspektivische Term für kleine ($\epsilon \to 0$) wie für große ϵ-Werte ($\epsilon \to \infty$) genauso verhält, wie der perspektivische Term, mit dem Unterschied, dass in dem paraperspektivischen Fall der Parameter ϵ linear eingeht.

Die Benutzung eines perspektivischen Kameramodelles bedeutet, dass die Formeln für die Projektion (siehe Abschnitt 5.3.3) unverändert in die Rekonstruktionsberechnung eingehen. Der oben beschriebene Faktor der perspektivischen Projektion $1/(1+\epsilon_i)$ wird also nicht modifiziert. Das perspektivische Modell geht oft einher mit teuren nichtlinearen Lösungs-/Minimierungsprozessen.

Affine Modelle relaxieren das perspektivische Modell mit linearen Approximationen, d.h. dass der oben beschriebene Faktor der perspektivischen Projektion $1/(1+\epsilon_i)$ (unter gewissen Einschränkungen) durch den para-perspektivischen Term ersetzt wird. Diese Modelle führen zu linearen Lösungsmethoden, deren Berechnungen meist nach nur wenigen Iterationen gegen die gesuchte Lösung konvergieren. Der Ansatz dieses Algorithmus ist es, die einfache Verbindung aus Gleichung 5.6 auszunutzen, die zwischen dem perspektivischen und seiner linearen Approximationen besteht, um auf die linearen Lösungsmethoden zurückzugreifen (siehe Abschnitt 5.4.2). Wenn Lösungen mit dem affinen Modell für das perspektivische Kameramodell berechnet worden sind, dann gehen diese Lösungen in die Berechnung des Endergebnisses unter Verwendung des perspektivischen Kameramodells wieder ein. Der Ansatz gilt für die Rekonstruktion von kalibrierten Kameras, d.h. Kameras, deren intrinsische Parameter bekannt sind.

Oft werden perspektivische Modelle mit affinen derart kombiniert, dass die affine Lösung als initiale Schätzung für die Lösung eines nichtlinearen Minimierungsprozesses benutzt wird. Allerdings gibt es keinen Beweis dafür, das die affine Lösung eine gute Initialisierung für einen solchen Prozess darstellt.

5.4.1 Eingabe

Das in dieser Projektgruppe entwickelte Programm greift auf Daten von den in Abschnitt 2.1.2 beschriebenen drei Kalibriergegenständen (Kalibriergitter, Kalibrierstab, Kalibrierquader) zurück. Diese Daten werden in verschiedenen Phasen des Programmes ausgewertet.

Die in der Kalibrierungsphase berechnete intrinsische Matrix, die Informationen über das Auflösungsvermögen und über das Bildseitenverhältnis der Videokamera beinhaltet, bildet eine Voraussetzung für eine euklidische dreidimensionale Rekonstruktion (siehe Abschnitt 3).

Wenn drei Bildsequenzen existieren, die die Voraussetzungen erfüllen, die in Abschnitt 2.1.2 beschrieben worden sind, kann jede der drei Sequenzen für sich dreidimensional rekonstruiert werden. Der Rekonstruktionsalgorithmus erwartet für jede der drei Sequenzen zwei Eingaben.

Die erste Eingabe für den Algorithmus besteht in den intrinsischen Parametern (in Form einer Matrix) der in dem C-Bogen verwendeten Videokamera. Diese Parameter müssen für jede der drei Aufnahmesequenzen gleich sein. Das ist der Fall, wenn derselbe C-Bogen für alle drei Sequenzen verwendet worden ist. Wenn der C-Bogen innerhalb der Aufnahmesequenzen ausgetauscht wird, liefert der Algorithmus eine Rekonstruktion, die vom Original abweicht. Aus diesem Grund ist es wichtig einen neuen C-Bogen zunächst mithilfe des Kalibriergitters zu kalibrieren, und diesen C-Bogen für alle drei Sequenzen zu verwenden.

Die zweite Eingabe für den Algorithmus besteht für jeden anatomischen Abschnitt aus je einer Matrix von zweidimensionalen korrespondierenden Punkten aus Pixelkoordinaten der zwölf abgebildeten Metallkugeln.

5.4.2 Die einzelnen Schritte des Algorithmus

Nachdem die Eingabe in einem vorbereitenden Schritt modifiziert wird, durchläuft der Rekonstruktionsalgorithmus mehrere Iterationen von Berechnungen bis ein Konvergenzkriterium den Abbruch der Schleife bestimmt. Jede Iteration umfasst folgende Schritte:

1. Aktualisierung der Projektion
2. Affine Rekonstruktion
3. Auswählen aus zwei Lösungen
4. Euklidische Rekonstruktion
5. Die Aktualisierung der extrinsischen Parameter und des Projektionsparameters ϵ_i

Am Ende können aus den zuletzt berechneten extrinsischen Parametern die Kamerapositionen ermittelt werden.

5.4.2.1 Vorbereitungen

Zunächst werden die Pixelkoordinaten, die aus der Segmentierung gewonnen worden sind, in das Bildkoordinatensystem überführt, indem die intrinsische Matrix invers auf die Pixelkoordinaten multipliziert wird. Die intrinsische Matrix braucht in den folgenden Berechnungen des Algorithmus nicht weiter berücksichtigt werden:

$$\tilde{\mathbf{m}}_B = \mathbf{A}^{-1}\tilde{\mathbf{m}}$$

Anschließend werden die Formeln für die Projektion (siehe Gleichung 5.4) in die Bildkoordinaten dahingehend umformuliert, dass der nichtlineare Projektionsterm durch die lineare Approximation ersetzt wird (siehe Gleichung 5.6):

$$\mathbf{m}_B = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \mathbf{I} \cdot \mathbf{M}_W + x_0 \\ \mathbf{J} \cdot \mathbf{M}_W + y_0 \end{pmatrix} \cdot (1 - \epsilon) \tag{5.7}$$

Damit ergeben sich für das para-perspektivische Kameramodell *neue* Formulierungen der Zeilenvektoren **I** und **J** der Bewegungsmatrix (im Vergleich zum perspektivischen Kameramodell in Gleichung 5.5):

$$\mathbf{I} = \frac{(\mathbf{i} - x_0\mathbf{k})^T}{t_z}, \qquad \mathbf{J} = \frac{(\mathbf{j} - y_0\mathbf{k})^T}{t_z} \tag{5.8}$$

Der darin enthaltene Projektionsparameter ϵ_i (in der Literatur als *perspective correction* bezeichnet) wird mit dem Wert Null initialisiert. Dadurch wird angenommen, dass alle betrachteten dreidimensionalen Punkte auf einer Ebene liegen und infolge der Iteration laufend neu korrigiert werden, weil der Projektionsparameter neu korrigiert wird. Mit jeder Iteration nehmen die Punkte der Form immer mehr die zu der realen Szene äquivalenten Positionen ein. Der erste Punkt $\tilde{\mathbf{M}}_{W0}$ bildet in der späteren Rekonstruktion den Weltkoordinatenursprung.

5.4.2.2 Die Aktualisierung der Projektion

Mit den Formeln der Projektion (siehe Gleichung 5.7) lassen sich die zweidimensionalen Punkte in Bildkoordinaten berechnen. In diese Formeln gehen der Projektionsparameter (in der affinen Approximation) und das Produkt der dreidimensionalen Punkte mit der Projektions-/Bewegungsmatrix ein. Diese Formeln werden nach dem Produkt aufgelöst:

$$(\mathbf{s}_{i,j}) = \begin{pmatrix} \mathbf{I}_j \cdot \mathbf{M}_{W,i} \\ \mathbf{J}_j \cdot \mathbf{M}_{W,i} \end{pmatrix} = \begin{pmatrix} (x_{i,j} - x_{0,j}) \cdot (1 + \epsilon_{i,j}) \\ (y_{i,j} - y_{0,j}) \cdot (1 + \epsilon_{i,j}) \end{pmatrix} \tag{5.9}$$

Diese Umformulierung der Projektion kann so verstanden werden, dass die zweidimensionalen Punkte (in Bildkoordinaten ausgedrückt und während der Schleife unverändert) in die Formeln der Projektion zur Berechnung der dreidimensionalen Punkte mit eingehen. Die Projektion wird also umgekehrt. Der Projektionsparameter beeinflusst die Berechnung, der entweder durch die Initialisierung mit Null (siehe Abschnitt 5.4.2.1) oder durch die vorherige Iteration einen neuen Wert besitzt. Das Ergebnis dieses Schrittes für k Bilder und n Punkte ist eine $(2k \times n)$-Matrix $\mathbf{S} = (\mathbf{s}_{i,j})$ (im Folgenden als measurement-Matrix bezeichnet) von aktualisierten Werten, die aus der umgekehrten Projektion herrühren.

5.4.2.3 Affine Rekonstruktion

Der nächste Schritt zur Berechnung der euklidischen Rekonstruktion besteht in der affinen Rekonstruktion. Hierbei wird eine Bewegungsmatrix $\mathbf{B}_{affine}$ und eine Form $\mathbf{F}_{affine}$ affin rekonstruiert, d.h. ohne Rücksicht auf Verzerrungen der Form und der Bewegung, die durch Skalierung und Scherung herrühren (affine Transformation). Durch eine affin rekonstruierte Kamera betrachtet, sieht die affin rekonstruierte Form aus, wie die zweidimensionale Vorlage.

Diese Affine Rekonstruktion wird aus der Singulärwertzerlegung der measurement Matrix $\mathbf{S} =$

$(\mathbf{s}_{i,j})$ des vorigen Schrittes (siehe Gleichung 5.9) gewonnen. Die Zerlegung ergibt drei Matrizen: eine Matrix $\mathbf{O}_1$ mit Linkssingularvektoren, eine Matrix $\mathbf{O}_2$ mit Rechtssingularvektoren und eine Diagonalmatrix $\mathbf{\Sigma}$ mit Singularwerten:

$$SVD(\mathbf{S}) = \mathbf{O}_1 \cdot \mathbf{\Sigma} \cdot \mathbf{O}_2 = \mathbf{S} \tag{5.10}$$

Für die dreidimensionale Rekonstruktion der Form und der Kamerapositionen wird gefordert, dass $\mathbf{F}_{affine}$ eine $3 \times n$-Matrix (dreidimensionale Punkte) und $\mathbf{F}_{affine}$ eine $2k \times 3$-Matrix ist (Projektionsmatrizen für dreidimensionale Punkte). Das kann nur berechnet werden, wenn die Diagonalmatrix $\mathbf{\Sigma}$ den Rang drei hat. Auch wenn $\mathbf{\Sigma}$ theoretisch die Bedingung erfüllt, kann aufgrund numerischer Instabilitäten der Rang größer als drei sein. Daher wird die Diagonalmatrix $\mathbf{\Sigma}$ durch eine andere Diagonalmatrix $\mathbf{\Sigma}'$ ersetzt, in der aber nur die drei größten Diagonalelemente/Singulärwerte aus der ursprünglichen Diagonalmatrix $\mathbf{\Sigma}$ übernommen werden.

$$\mathbf{S} = \underbrace{\mathbf{O}_1 \cdot \mathbf{\Sigma}' \cdot \mathbf{O}_2}_{\text{hat 3 größte Singulärwerte}} + \underbrace{\mathbf{O}_1 \cdot \mathbf{\Sigma}'' \cdot \mathbf{O}_2}_{\text{hat restliche Einträge}} \approx \mathbf{O}_1 \cdot \mathbf{\Sigma}' \cdot \mathbf{O}_2 \tag{5.11}$$

Diese neue Matrix $\mathbf{\Sigma}'$ wird in zwei Diagonalmatrizen derart zerlegt, dass z.B. beide Matrizen nur die Wurzeln der drei Singularwerte besitzen. Je eine Diagonalmatrix $\mathbf{\Sigma}^{1/2}$ wird auf eine Singularvektormatrix multipliziert und das ergibt letztenendes die zwei Matrizen $\mathbf{B}_{affine}$ und $\mathbf{F}_{affine}$, die die affine Rekonstruktion darstellen:

$$\mathbf{S} = \underbrace{(\mathbf{O}_1 \cdot \mathbf{\Sigma}^{1/2})}_{\mathbf{B}_{affine}} \cdot \underbrace{(\mathbf{\Sigma}^{1/2} \cdot \mathbf{O}_2)}_{\mathbf{F}_{affine}} \tag{5.12}$$

Zusammenfassend kann die measurement-Matrix als Produkt dieser beiden Matrizen aufgefasst werden,

$$\begin{pmatrix} \mathbf{s}_{11} & \dots & \mathbf{s}_{1n} \\ \vdots & \ddots & \vdots \\ \mathbf{s}_{k1} & \dots & \mathbf{s}_{kn} \end{pmatrix} = \begin{pmatrix} \mathbf{B}_1 \\ \vdots \\ \mathbf{B}_k \end{pmatrix}_{affine} \cdot \begin{pmatrix} \mathbf{F}_1 & \dots & \mathbf{F}_n \end{pmatrix}_{affine}$$

wobei sie entweder aus dem Produkt von zwei positiven Matrizen oder aus zwei negativen Matrizen berechnet werden kann.

$$\mathbf{S} = \mathbf{B} \cdot \mathbf{F} = (-\mathbf{B}) \cdot (-\mathbf{F}) \tag{5.13}$$

Das bedeutet, dass für die umgekehrte Projektion zwei mögliche Lösungen existieren, und zwar eine rekonstruierte Menge von Kameras und Punkten oder die gespiegelte Version davon. Diese Mehrdeutigkeit muss in einem weiteren Schritt aufgelöst werden, indem überprüft wird, welche von beiden Lösungen nicht zur originalen Form gespiegelt ist.

5.4.2.4 Auswählen aus zwei Lösungen (Resolving the Reversal Ambiguity)

Diese Zweideutigkeit lässt sich nur auflösen, indem alle möglichen Lösungswege verfolgt werden. In der ersten Iteration werden beide Lösungen gespeichert. In jeder folgenden Iteration werden von beiden Lösungen jeweils die beiden Folgelösungen berechnet und jeweils eine ausgewählt (siehe Abb. 5.3). Die Auswahl einer von zwei möglichen Lösungen geschieht in dieser

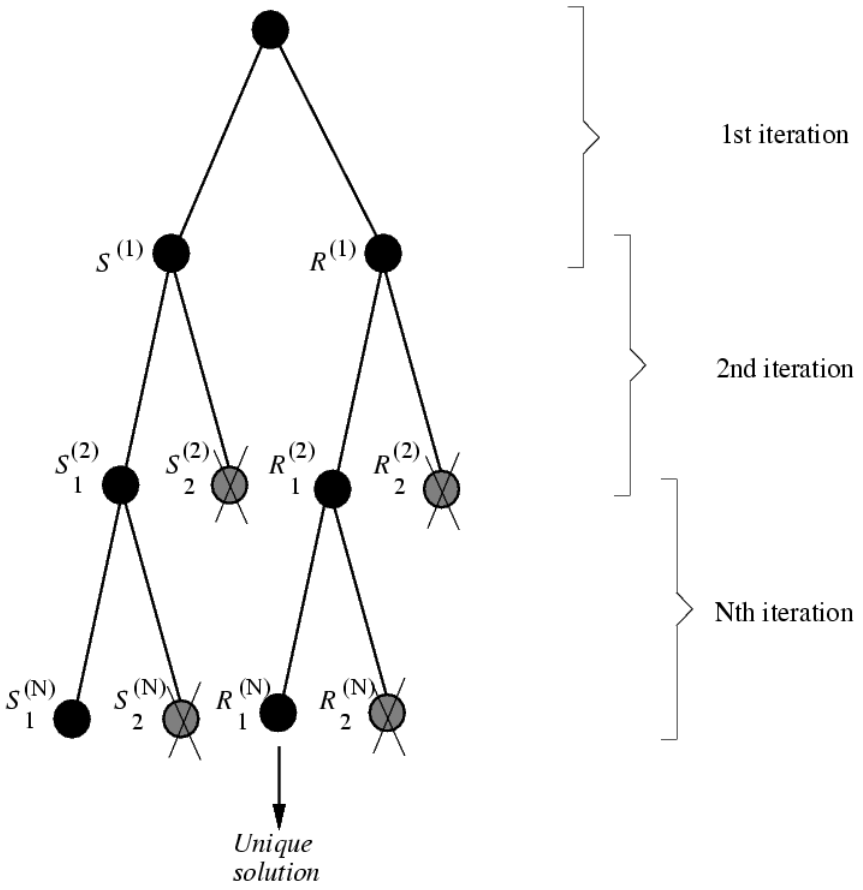


Abbildung 5.3: Es werden bei jeder Iteration je zwei Lösungen ausgewählt und am Schluss eine eindeutige Lösung bestimmt [CH94]

Implementierung unter Einbeziehung des Domänenwissens über die originale Form des Kalibrierquaders. Anhand der Händigkeiten von Koordinatensystemen, die die Kanten der rekonstruierten Form einerseits und der originalen Form andererseits bilden, kann entschieden werden, ob die rekonstruierte Form zu der Originalen gespiegelt ist oder nicht. Es wird ebenfalls entschieden, ob die Vorgängerform zur Originalen gespiegelt ist oder nicht. Dann wird immer die Lösung ausgewählt, die zur Vorgängerform nicht gespiegelt ist.

5.4.2.5 Euklidische Rekonstruktion

Die affine Rekonstruktion liefert einen Satz von dreidimensionalen Punkten und Projektionsmatrizen in Form von zwei Matrizen, $\mathbf{B}_{affine}$ (affine Bewegungsmatrizen) und $\mathbf{F}_{affine}$ (affine Form). Diese beiden Matrizen sind für die Problemstellung noch nicht als Lösung geeignet, denn sie sind nicht euklidisch rekonstruiert (die Form und die Matrizen resultieren nicht aus einer Transformation starrer Körper). Die Suche nach einer Lösung beschränkt sich auf die Berechnung einer Matrix **T**, die zwei Anforderungen erfüllen soll. Diese Matrix soll einerseits von rechts an die affin rekonstruierten Projektionsmatrizen (motion) multipliziert werden, um die euklidisch

rekonstruierten Projektionsmatrizen zu erhalten:

$$\mathbf{B}_{rigid} = \mathbf{B}_{affine} \cdot \mathbf{T} \tag{5.14}$$

Andererseits soll die inverse Multiplikation von links an die affin rekonstruierte Form die euklidisch rekonstruierten Punkte ergeben:

$$\mathbf{F}_{rigid} = \mathbf{T}^{-1} \cdot \mathbf{F}_{affine} \tag{5.15}$$

Aus den euklidisch rekonstruierten Projektionsmatrizen ergeben sich die gesuchten extrinsischen Parameter (siehe Abschnitt 5.4.2.6) und daraus die gesuchten Kamerapositionen (siehe Abschnitt 5.4.3).

Um eine solche Matrix **T** zu erhalten, werden drei Bedingungen an die euklidische Rekonstruktion der Kameras gestellt. Diese Bedingungen werden in Form von Gleichungen ausgedrückt. Wenn die Projektionsmatrizen der Kameras eine euklische Rekonstruktion darstellen sollen, dann müssen die aus ihnen resultierenden extrinsischen Parameter aus einer Orthonormal-Basis und einem Verschiebungsvektor zum Weltkoordinatenursprung in Kamerakoordinaten bestehen. Die Orthonormalbasis wiederum besteht aus drei zueinander orthogonalen Einheitsvektoren, von denen einer die Blickrichtung vorgibt und die beiden anderen die Bildebene aufspannen. Diese Basis bildet am Ende die Rotationsmatrix $\mathbf{R}_{3\times3}$ der extrinsischen Matrix (siehe Gleichung 5.3.3). Zunächst hat die affine Rekonstruktion Projektionsmatrizen $\mathbf{B}_{affine,j}$ geliefert, die aus nur zwei Zeilen $\mathbf{I}_{affine,j}$ und $\mathbf{J}_{affine,j}$ bestehen. Die erste Bedingung an eine euklidische Rekonstruktion der Projektionsmatrizen $\mathbf{B}_{rigid,j}$ ist, dass die quadratischen Normen ihrer beiden Zeilen $\mathbf{I}_{rigid,j}$ und $\mathbf{J}_{rigid,j}$ gleich groß sein müssen. Das hat den Hintergrund, dass alle Basisvektoren der extrinsischen Matrix Einheitsvektoren und damit alle gleich lang sind. Eine Zeile der extrinsischen Matrix enthält zusätzlich zu einem dreidimensionalen Basisvektor einen Translationsanteil, daher beträgt die quadratische Norm $1 + x_0^2$ bzw. $1 + y_0^2$ (eine 1 wegen der Norm des Basisvektors und x_0^2 bzw. y_0^2 wegen dem Translationsanteil). In den folgenden Bedingungen werden die Normen der Zeilen der Bewegungsmatrix zu den Normen der entsprechenden Zeilen der extrinsischen Matrix mithilfe des Strahlensatzes in Beziehung gebracht. Damit verhält sich die Norm von I zu der von J genauso wie die Norm $1 + x_0^2$ zu $1 + y_0^2$. Es ergeben sich für k Bilder ebensoviele Bedingungen der Form:

$$\frac{\|\mathbf{I}_{rigid,j}\|^2}{1 + x_{0,j}^2} = \frac{\|\mathbf{J}_{rigid,j}\|^2}{1 + y_{0,j}^2} \quad \Longrightarrow \quad \frac{\mathbf{I}_{affine,j}\mathbf{T}\mathbf{T}^T\mathbf{I}_{affine,j}^T}{1 + x_{0,j}^2} - \frac{\mathbf{J}_{affine,j}\mathbf{T}\mathbf{T}^T\mathbf{J}_{affine,j}^T}{1 + y_{0,j}^2} = 0 \tag{5.16}$$

Die zweite Bedingung an die euklidische Rekonstruktion der Kameras fordert die Orthogonalität der Basisvektoren der extrinsischen Matrix. Diese ergibt sich, wenn das Skalarprodukt beider Zeilen der Projektionsmatrix, die euklidisch rekonstruiert wird, nicht mehr als das Produkt der beiden Bildkoordinaten x_0 und y_0 des gewählten Bezugspunktes der Form ergibt. Damit ergeben sich neben den Bedingungen aus Gleichung 5.16 für alle k Bilder insgesamt k zusätzliche Bedingungen der Form:

$$\begin{aligned} \mathbf{I}_{rigid,j} \cdot \mathbf{J}_{rigid,j} &= \frac{x_{0,j}y_{0,j}}{2}\left(\frac{\|\mathbf{I}_{rigid,j}\|^2}{1 + x_{0,j}^2} + \frac{\|\mathbf{J}_{rigid,j}\|^2}{1 + y_{0,j}^2}\right) \quad \Longrightarrow \\ \mathbf{I}_{affine,j}\mathbf{T}\mathbf{T}^T\mathbf{J}_{affine,j}^T &- \left(\frac{\mathbf{I}_{affine,j}\mathbf{T}\mathbf{T}^T\mathbf{I}_{affine,j}^T}{1 + x_{0,j}^2} + \frac{\mathbf{J}_{affine,j}\mathbf{T}\mathbf{T}^T\mathbf{J}_{affine,j}^T}{1 + y_{0,j}^2}\right) = 0 \end{aligned} \tag{5.17}$$

Dadurch ergibt sich insgesamt ein homogenes Gleichungssystem mit $2k$ Gleichungen. Damit die triviale Lösung verhindert wird, ist eine weitere Bedingung erforderlich.

Die dritte Bedingung fordert die Orthonormalität der Basisvektoren, indem festgelegt wird, dass die quadratische Norm von je einem der beiden Zeilenvektoren der Projektionsmatrix nur um $x_{0,1}^2$ bzw. $y_{0,1}^2$ von dem Wert 1 abweichen darf:

$$\|\mathbf{I}_{\textit{rigid},1}\|^2 = 1 + x_{0,1}^2 \quad \Longrightarrow \quad \mathbf{I}_{\textit{affine},1}\mathbf{T}\mathbf{T}^T\mathbf{I}_{\textit{affine},1}^T = 1 + x_{0,1}^2$$

oder

$$\|\mathbf{J}_{\textit{rigid},1}\|^2 = 1 + y_{0,1}^2 \quad \Longrightarrow \quad \mathbf{J}_{\textit{affine},1}\mathbf{T}\mathbf{T}^T\mathbf{J}_{\textit{affine},1}^T = 1 + y_{0,1}^2 \tag{5.18}$$

Alle diese Bedingungen werden zu einem linearen Gleichungssystem mit $2k + 1$ Gleichungen zusammengefasst, dessen Unbekannte die Matrix **T** ist. Das Gleichungssystem hängt allerdings nichtlinear von **T** ab. Daher wird das Matrixprodukt $\mathbf{T}\mathbf{T}^T$ mit der Matrix **Q** substituiert. Die Matrix **Q** wird anschließend als sechsdimensionaler Vektor **q** umformuliert (sechs Dimensionen weil **Q** (positiv) symmetrisch ist). Insgesamt ergibt das ein lineares Gleichungssystem mit $2k + 1$ Gleichungen in 6 Unbekannten (daher kann erst für mindestens $k = 3$ Bilder eine Lösung berechnet werden), dessen Lösung mithilfe der Singulärwertzerlegung approximiert wird. Der Lösungsvektor **q** wird dann wieder zu einer Matrix **Q** umformuliert und mithilfe der Eigenwert-Zerlegung zerlegt:

$$\textit{Eigenvalue}(\mathbf{Q}) = \mathbf{O} \cdot \mathbf{D} \cdot \mathbf{O}^T = \mathbf{Q} \tag{5.19}$$

Die Matrix **O** enthält dabei die Eigenvektoren von **Q** und die Diagonalmatrix **D** enthält die Eigenwerte in absteigender Reihenfolge. Wenn **D** in zwei Matrizen $\mathbf{D}^{1/2}$ und $\mathbf{D}^{1/2}$ zerlegt wird, die die Wurzeln der Eigenwerte als Diagonalelemente besitzen, und je eine an eine Eigenvektormatrix **O** und $\mathbf{O}^T$ multipliziert wird, dann ergeben sich die gesuchten Matrizen **T** und $\mathbf{T}^T$:

$$\mathbf{Q} \;=\; \underbrace{(\mathbf{O} \cdot \mathbf{D}^{1/2})}_{\mathbf{T}} \cdot \underbrace{(\mathbf{D}^{1/2} \cdot \mathbf{O}^T)}_{\mathbf{T}^T} = \mathbf{T} \cdot \mathbf{T}^T \tag{5.20}$$

5.4.2.6 Die Aktualisierung der extrinsischen Parameter und des perspektivischen Korrekturparameters ϵ_i

Da die Matrix **T** nun bekannt ist, wird diese auf die beiden Zerlegungsmatrizen $\mathbf{B}_{\textit{affine}}$ und $\mathbf{F}_{\textit{affine}}$ der affinen Rekonstruktion multipliziert (siehe Abschnitt 5.4.2.5). Dies ergibt eine euklidische Rekonstruktion der Projektionsmatrizen $\mathbf{B}_{\textit{rigid}}$ und der Form $\mathbf{F}_{\textit{rigid}}$. Aus den zwei Zeilen jeder Projektionsmatrix werden anschließend die extrinsischen Parameter berechnet, um daraufhin die Projektionen für die nächste Iteration zu aktualisieren.Die Aktualisierung umfasst die Neuberechnung aller drei Koordinaten t_x, t_y und t_z des Verschiebungsvektors **t** und die Berechnung der Rotationsmatrix $\mathbf{R}_{3\times3}$, die die drei Zeilenvektoren $\mathbf{i}^T, \mathbf{j}^T$ und $\mathbf{k}^T$ beinhaltet.

Für den Translationsvektor **t** muss zuerst die letzte Koordinate t_z berechnet werden bevor die

ersten beiden Koordinaten t_x und t_y berechnet werden können:

$$\begin{aligned} t_{z,j} &= \frac{1}{2}\left(\frac{\sqrt{1+x_{0,j}^2}}{\|\mathbf{I}_{rigid}\|}+\frac{\sqrt{1+y_{0,j}^2}}{\|\mathbf{J}_{rigid}\|}\right) \\ t_{x,j} &= x_0 \cdot t_{z,j} \\ t_{y,j} &= y_0 \cdot t_{z,j} \end{aligned} \tag{5.21}$$

Für die Rotationsmatrix muss der letzte Zeilenvektor $\mathbf{k}^T$ (Blickrichtungsvektor der Kamera) vor der Berechnung der ersten beiden Zeilenvektoren $\mathbf{i}^T$ und $\mathbf{j}^T$ (Spannvektoren der Bildebene der Kamera) ermittelt werden. Dies geschieht mithilfe der para-perspektivisch modifizierten Zeilenvektoren $\mathbf{I}_{rigid}$ und $\mathbf{J}_{rigid}$ der Bewegungsmatrix $\mathbf{B}_{rigid}$ (siehe Gleichung 5.8):

$$\begin{aligned} \mathbf{k}_j &= \mathbf{W}^{-1} \cdot t_{z,j}^2 (\mathbf{I}_{rigid,j}^T \times \mathbf{I}_{rigid,j}^T) \\ \mathbf{i}_j &= t_{z,j}\mathbf{I}_{rigid,j}^T + x_{0,j}\mathbf{k}_j \\ \mathbf{j}_j &= t_{z,j}\mathbf{J}_{rigid,j}^T + y_{0,j}\mathbf{k}_j \end{aligned} \tag{5.22}$$

mit

$$\mathbf{W} = \mathbf{E}_{3\times 3} - t_{z,j}y_{0,j} \cdot \mathbf{Cross}(\mathbf{I}_{rigid,j}) + t_{z,j}x_{0,j} \cdot \mathbf{Cross}(\mathbf{J}_{rigid,j})$$

wobei

$$\mathbf{E}_{3\times 3} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

die Einheitsmatrix ist und

$$\mathbf{Cross}(\mathbf{u}) = \begin{pmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{pmatrix}$$

die Matrix-Formulierung des Kreuzproduktes ist, mit dem Effekt, dass

$$\mathbf{Cross}(\mathbf{u}) \cdot \mathbf{v} = \mathbf{u} \times \mathbf{v}$$

All diese Berechnungen gelten unter der Annahme, dass der Blickrichtungsvektor **k** das Resultat des Kreuzproduktes der beiden Bildebenen-Spannvektoren **i** und **j** ist:

$$\mathbf{k}_j = \mathbf{i}_j \times \mathbf{j}_j$$

5.4.3 Ausgabe

Die Schleife, die in Abschnitt 5.4.2 beschrieben wurde, wird durch ein Konvergenzkriterium abgebrochen, das die Differenz zwischen den neuberechneten ϵ-Werten der aktuellen Iteration zu den Werten der vorhergehenden Iteration berechnet. Das Kriterium feuert, wenn die Differenz

eine vordefinierte Schwelle unterschreitet. Die zuletzt berechneten ϵ-Werte und damit die zuletzt berechneten extrinsischen Parameter bilden die Ausgabe des Algorithmus. Daraus werden die Kamerapositionen bzw. optische Projektionszentren C_j bestimmt.

Das optische Projektionszentrum $\mathbf{C}_{3,j}$ bildet den Koordinatenursprung des Kamerakoordinatensystems in Weltkoordinaten (siehe Abschnitt 5.3.3). Multipliziert mit den extrinsischen Koordinaten des entsprechenden Bildes j ergibt das Produkt Null:

$$\mathbf{D}_{4\times4,j} \cdot \mathbf{C}_{4,j} = \begin{pmatrix} \mathbf{R}_{3\times3} & \mathbf{t}_3 \\ \mathbf{0}_3^T & 1 \end{pmatrix}_j \begin{pmatrix} \mathbf{C}_{3,j} \\ 1 \end{pmatrix} = \mathbf{R}_{3\times3,j} \cdot \mathbf{C}_{3,j} + \mathbf{t}_{3,j} = \mathbf{0}$$

Dieses Gleichungssystem wird nach dem gesuchten optischen Projektionszentrum $\mathbf{C}_{3,j}$ aufgelöst:

$$\mathbf{C}_{3,j} = -\mathbf{R}_{3\times3,j}^{-1} \cdot \mathbf{t}_{3,j} \tag{5.23}$$

Diese Gleichung macht deutlich, dass das optische Projektionszentrum $\mathbf{C}_{3,j}$ in Weltkoordinaten dem Verschiebungsvektor $\mathbf{t}_{3,j}$ in Kamerakoordinaten entspricht.

Damit ergeben sich für k Bilder ebensoviele optische Kamerazentren. Die n dreidimensional rekonstruierten Punkte bilden eine weitere Ausgabe, die in der dreidimensionalen Ansicht des Programmes dargestellt wird, sodass ein visueller Eindruck der Rekonstruktionsqualität gewonnen werden kann. Anhand der Abweichungen der rekonstruierten Punkte zu den theoretischen kann das Programm die Güte der Rekonstruktion quantitativ analysieren (siehe Abschnitt 5.6).

Aus der Kenntnis der Kamerapositionen können nun zusätzliche Daten dreidimensional rekonstruiert werden, wie z.B. der Kalibrierstab, die Achsenendpunkte des Knochens, sowie die der Prothese.

5.5 Dreidimensionale Rekonstruktionen für beliebige, korrespondierende Punkte

Nach der euklidischen dreidimensionalen Rekonstruktion des Kalibrierquaders und der Kamerapositionen können aus denselben Bildern beliebige korrespondierende Punkte rekonstruiert werden, auch wenn für die kein Kalibrierobjekt existiert. Diese Rekonstruktionen können deshalb durchgeführt werden, weil die Kamerapositionen und deren extrinsische Parameter (Blickrichtung, Bildebene) bekannt sind.

Die Eingabe ist die Folge von zweidimensionalen Punkten aus Pixelkoordinaten der abgebildeten Stabmarkierung, der in einem Bild noch sichtbaren oberen und unteren Endpunkte des Kalibrierstabes, der Achsenendpunkte des Knochens und der der Prothese. Diese Informationen werden nur in einem, dem euklidischen Rekonstruktionsalgorithmus nachfolgenden Schritt vom Programm zur Berechnung der dreidimensionalen Achsen benötigt. Wenn eine euklidische Rekonstruktion des Kalibrierdiamanten erfolgt ist, wird diese Eingabe anhand der aus der euklidischen Rekonstruktion bekannten Kamerapositionen und -parametern ebenfalls dreidimensional rekonstruiert. Dazu stellt das Programm zwei Algorithmen bereit, die in ihrem Aufbau sehr ähnlich zueinander sind.

Zunächst erhalten beide Algorithmen die Pixelkoordinaten und die Indizes der Bilder, zu denen die Pixelkoordinaten gehören. In einem vorbereitenden ersten Schritt werden die Koordinaten der Pixel mithilfe der berechneten intrinsischen und extrinsischen Matrizen in das Weltkoordinatensystem überführt, sodass für je ein Bild die Differenzvektoren von einer Kameraposition zu diesen Punkten berechnet werden können. Diese Vektoren bilden die Richtung für Sicht-Geraden oder zusammen mit einem anderen Vektor desselben Bildes eine Ebene.

Der eine Algorithmus rekonstruiert dreidimensional einzelne Punkte, indem Sicht-Strahlen der Pixel korrespondierender Punkte miteinander geschnitten werden. Solche Geraden sind allerdings aufgrund der Ungenauigkeiten der Segmentierung mit hoher Wahrscheinlichkeit windschief zueinander. Daher werden zu je einem Paar von Sicht-Geraden die Lotfußpunkte der beiden Geraden berechnet und aus diesen beiden Punkten der Schwerpunkt gebildet. Der Algorithmus berechnet die Schwerpunkte für alle paarweisen Kombinationsmöglichkeiten von Sicht-Geraden und bildet dann den Schwerpunkt aus allen Resultaten.

Der andere Algorithmus rekonstruiert ganze Geraden, indem für alle Bilder je eine eine Ebene, die sich für ein Bild aus dem Kamerazentrum und aus zwei Punkten ergibt, mit den Sicht-Geraden aller anderen Bilder geschnitten wird. Für jede Ebene häufen sich Schnittpunkte an zwei Bereichen der Ebene. In beiden Bereichen werden diese Schnittpunkte gemittelt, sodass sich zwei dreidimensionale Punkte ergeben, die die Endpunkte der rekonstruierten Geraden ergeben.

Um auch eine quantitative Unsicherheit angeben zu können, wird bei beiden Algorithmen mithilfe der Gesetze der Fehlerrechnung der Fehler der Mittelwerte berechnet. In der zugrunde liegenden Implementierung wird die empirische mittlere absolute Abweichung $|\bar{\Delta x}|_{\bar{x}}$ als Streuungsparameter berechnet. Dafür wird zunächst der arithmetische Mittelwert $\bar{x}$ der Stichprobe berechnet und dann die Abweichung:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad\qquad |\bar{\Delta x}|_{\bar{x}} = \frac{1}{N}\sum_{i=1}^{N} |x_i - \bar{x}| \tag{5.24}$$

5.6 Qualität der Rekonstruktion

Basierend auf Ergebnissen mit synthetisch generierten Daten und Phantomdaten (Ober- und Unterschenkel aus Polyurethan, inkl. Prothese) konnte nachgewiesen werden, dass der Rekonstruktionsalgorithmus in der Lage ist, Punkte, Achsen und Kamerapositionen exakt zu rekonstruieren. Der absolute Positionsfehler liegt dabei im Bereich von $10^{-6}m$.
Um die Robustheit des Algorithmus auf verrauschten Daten zu testen, wurden die Eingabedaten (Pixelkoordinaten) in einer dafür erstellten Testumgebung (siehe Abbildung 5.4) mit gleichverteiltem additiven Rauschen versehen. Dabei wurden unterschiedliche Konfigurationen untersucht, bei denen die Anzahl der verwendeten Aufnahmen, die Positionen der Kameras sowie das Rauschen variiert worden sind. In den Tabellen 5.1 bis 5.4 befinden sich die Ergebnisse für drei, vier, sieben und zwölf Aufnahmen. Jede Zeile repräsentiert die zusammengefassten Ergebnisse einer Testreihe mit einigen hundert Konfigurationen für den in der ersten Spalte der Tabelle angegebenen Rauschwert (in Pixeln). Die Kameras befanden sich dabei auf Kreisbögen verschiedener Radien um die aufgenommene Szene. Aufgezeichnet wurden Winkel- und Punktabweichungen.

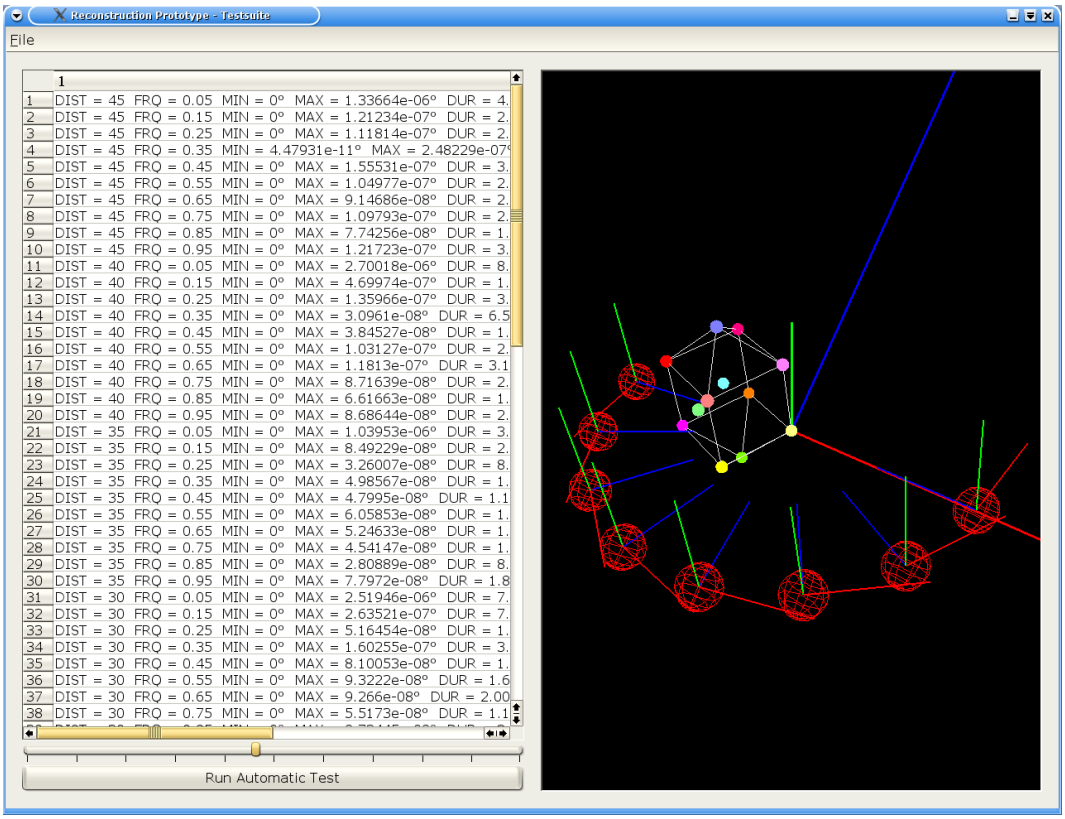


Abbildung 5.4: Testumgebung für die Rekonstruktion

Zur Berechnung der Winkelabweichung wurden alle Raumwinkel, die innerhalb des rekonstruierten und originalen Kalibrierdiamanten auftreten, berechnet und verglichen. Die minimalen, durchschnittlichen und maximalen Winkeldifferenzen sind in den ersten drei Spalten (min.W., mittl.W. und max.W.) der Tabellen aufgeführt. Die in den nächsten beiden Spalten (mittl.P. und max.P.) angegebenen Punktabweichungen sind die auf den Einheitswürfel bezogenen relativen Abweichungen der rekonstruierten Diamantenpunkte von den Originalpunkten. Die letzte Spalte gibt die Erfolgsquote der Rekonstruktionen an. Dabei bedeutet ein Fehlschlag, dass der Algorithmus nicht konvergiert. Dies ist in den meisten Fällen auf redundante Bildinformationen zurückzuführen, die aufgrund ungünstiger Kamerapositionen entstehen können. Insgesamt wurden über 10.000 verschiedene Konfigurationen getestet.

5.7 Ausrichtung der Rekonstruktionsergebnisse

Bei jedem einzelnen der drei rekonstruierten Bereiche (Hüfte, Knie, Knöchel) wurde die Geometrie so rekonstruiert, dass ein bestimmter Diamantpunkt im Ursprung zu liegen kommt. Da der Diamant allerdings zwischen den Aufnahmesequenzen verschoben und eventuell gedreht werden könnte, müssen die unabhängig von einander rekonstruierten Szenen wiederum so gedreht und verschoben werden, dass sie sich in der korrekten Lage befinden. Außerdem ist eine korrekte Skalierung jeder Szene erforderlich, da die Rekonstruktion nur bis auf einen skalaren Faktor genau bestimmt ist. Hierzu wird ein separater Ausrichtungsalgorithmus verwendet, der in folgenden

Rauschen	min.W.	mittl.W.	max.W.	mittl.P.	max.P.	Quote
0	2.241e-12	4.321e-08	1.932e-07	1.079e-06	4.129e-06	88.78%
1	9.384e-06	0.01633	0.07774	0.05233	0.1445	85.61%
2	2.034e-05	0.0321	0.1596	0.1308	0.37	84.88%
3	3.062e-05	0.04579	0.2311	0.2794	0.7458	85.12%
5	4.614e-05	0.06862	0.3513	0.3576	0.9568	83.17%
10	8.267e-05	0.1179	0.6148	0.7349	2.003	79.02%
20	0.0001332	0.1975	1.049	1.379	4.285	65.12%

Tabelle 5.1: Rekonstruktionsergebnisse mit 3 Kameras

Rauschen	min.W.	mittl.W.	max.W.	mittl.P.	max.P.	Quote
0	1.199e-11	1.166e-07	4.293e-07	9.033e-07	3.046e-06	96.83%
1	1.117e-05	0.01542	0.07155	0.0667	0.2337	94.88%
2	1.796e-05	0.02729	0.1346	0.1627	0.4804	92.93%
3	2.611e-05	0.0384	0.1926	0.2528	0.7732	92.93%
5	3.949e-05	0.05874	0.3017	0.4392	1.218	91.95%
10	6.911e-05	0.1055	0.555	1.054	3.132	87.8%
20	0.0001201	0.1743	0.9309	1.59	4.181	74.63%

Tabelle 5.2: Rekonstruktionsergebnisse mit 4 Kameras

Rauschen	min.W.	mittl.W.	max.W.	mittl.P.	max.P.	Quote
0	5.071e-12	8.038e-08	3.037e-07	6.131e-07	2.189e-06	99.51%
1	6.999e-06	0.01168	0.05485	0.05429	0.1607	98.29%
2	1.21e-05	0.01999	0.1012	0.1447	0.4322	97.07%
3	1.954e-05	0.02949	0.1516	0.2364	0.6449	96.83%
5	2.721e-05	0.04284	0.223	0.406	1.158	95.12%
10	5.485e-05	0.07788	0.4116	0.7425	2.122	92.68%
20	9.14e-05	0.1432	0.7482	1.569	4.781	87.8%

Tabelle 5.3: Rekonstruktionsergebnisse mit 7 Kameras

Rauschen	min.W.	mittl.W.	max.W.	mittl.P.	max.P.	Quote
0	2.764e-12	3.916e-08	1.671e-07	0.00953	0.05377	99.76%
1	5.007e-06	0.007795	0.03544	0.04266	0.1417	99.02%
2	8.1e-06	0.01427	0.06707	0.1026	0.3074	97.56%
3	1.1e-05	0.01807	0.09191	0.1762	0.5156	96.83%
5	2.006e-05	0.029	0.1507	0.3196	0.8817	96.59%
10	3.24e-05	0.05237	0.2783	0.5689	1.614	94.39%
20	6.533e-05	0.09626	0.5227	1.009	2.869	91.46%

Tabelle 5.4: Rekonstruktionsergebnisse mit 12 Kameras

Schritten vorgeht:

1. Bestimmung des Skalierungsfaktors für jede der drei Szenen
2. Ausrichtung jeder der Szenen an der y-Achse
3. Korrektur der Abstände von Hüfte zu Knie bzw. Knie zu Knöchel
4. Ausrichtung der Sequenzen an einer Ebene

Eine schematische Darstellung der einzelnen Schritte des Algorithmus findet sich in den Abbildung 5.5 bis 5.8.

Bestimmung des Skalierungsfaktors für jede der drei Szenen

Der korrekte Skalierungsfaktor lässt sich aus den Distanzen zwischen den rekonstruierten Diamantpunkten ableiten, da die tatsächlichen Abstände zwischen den Punkten bekannt sind. Um eine robuste Schätzung zu erhalten, werden bei der Berechnung des Skalierungsfaktors alle 66 verschiedenen Distanzen zwischen den zwölf rekonstruierten Punkten berechnet und zu den tatsächlichen in Relation gesetzt.
Werden die rekonstruierten Punkte mit P_i^r und die Originalpunkte mit P_i für $i = 1, \cdots, 12$ bezeichnet, ergibt sich folgender Skalierungsfaktor

$$s = \frac{1}{66} \sum_{1 \leq i \leq 12, i \neq j} \frac{\|P_i - P_j\|}{\|P_i^r - P_j^r\|}$$

und damit die Skalierungsmatrix

$$\mathbf{S} = \begin{pmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & s & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

für homogene Koordinaten. Das Ergebnis nach Anwendung der Skalierungsmatrix wird in Abbildung 5.9 dargestellt.

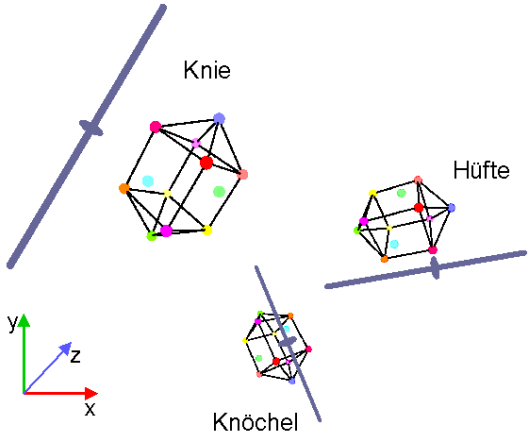


Abbildung 5.5: Das Ergebnis der Rekonstruktion

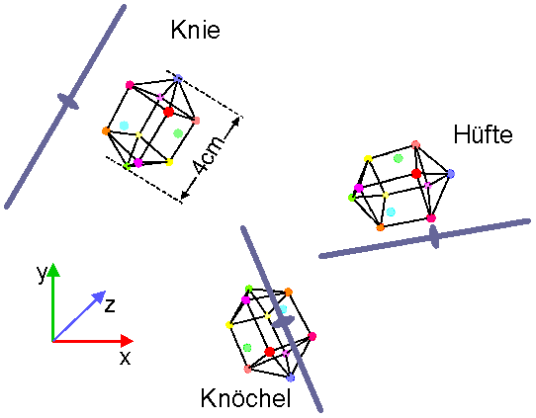


Abbildung 5.6: Gleichmäßige Skalierung der Diamanten

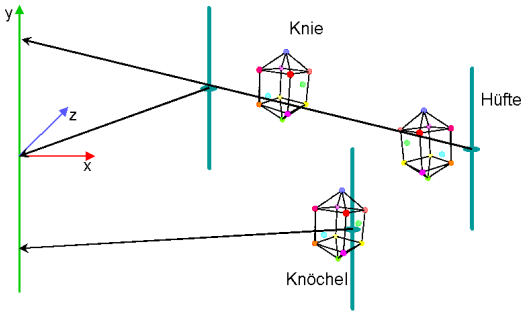


Abbildung 5.7: Ausrichten der Diamanten an der y-Achse

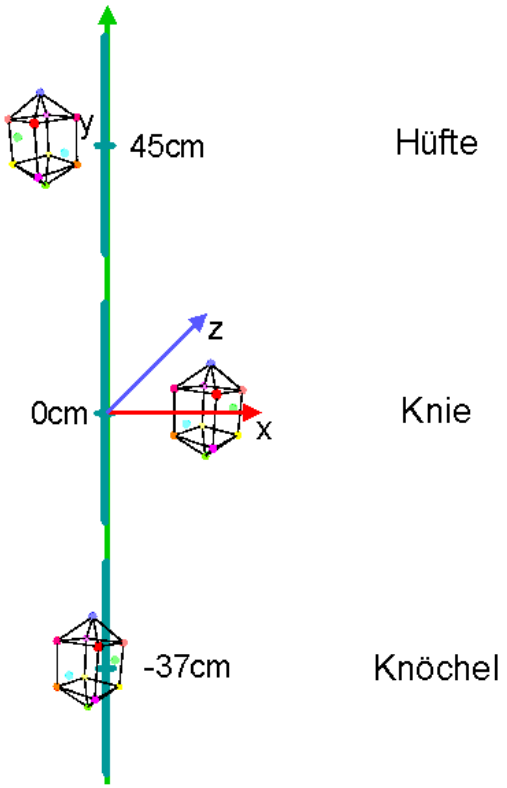


Abbildung 5.8: Ausrichten entlang der y-Achse

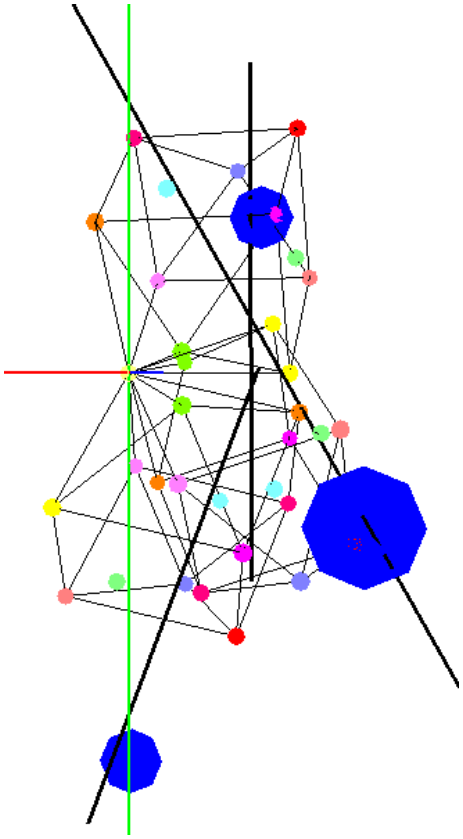


Abbildung 5.9: Das Ergebnis der Rekonstruktion nach der Skalierung

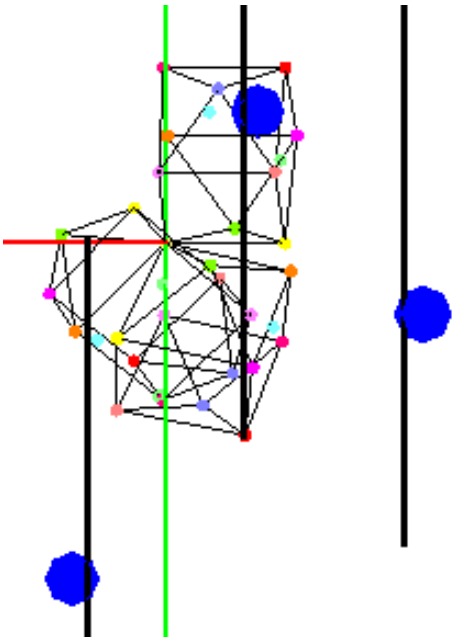


Abbildung 5.10: Nach der Rotation um die y-Achse in die yz-Ebene

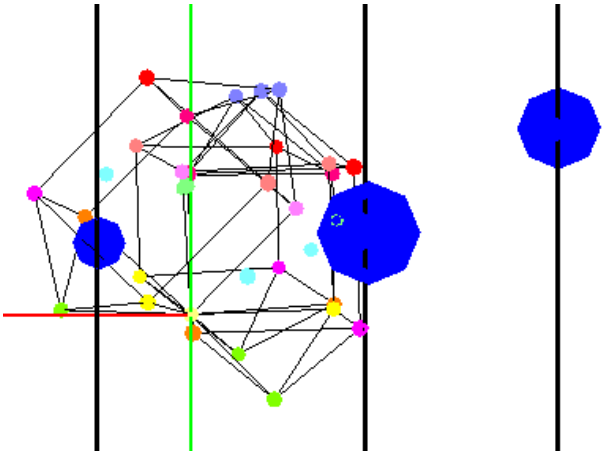


Abbildung 5.11: Ergebnis der Rotation um die x- auf die y-Achse

Ausrichtung jeder der Szenen an der y-Achse

In diesem Schritt wird jede der rekonstruierten Szenen so gedreht, dass der Kalibrierungsstab parallel zur y-Achse liegt. Die durchzuführende Rotation teilt sich folgendermaßen in zwei Teilrotationen auf: $\mathbf{R}_{ges} = \mathbf{R}_x\mathbf{R}_y$. Dabei wird die Szene zunächst mittels $\mathbf{R}_y$ um die y-Achse in die yz-Ebene (vergleiche Abbildung 5.10) und anschließend um die x-Achse auf die y-Achse gedreht (siehe Abbildung 5.11). Wird der Richtungsvektor des Stabes mit $\mathbf{d} = (d_x, d_y, d_z)$ bezeichnet, ergibt sich als Rotationswinkel für $\mathbf{R}_y$

$$\alpha_y = \arccos \frac{d_z}{\|\mathbf{d}\|}$$

und somit

$$\mathbf{R}_y = \begin{pmatrix} \cos\alpha_y & 0 & -\sin\alpha_y & 0 \\ 0 & 1 & 0 & 0 \\ \sin\alpha_y & 0 & \cos\alpha_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Analog ergibt sich

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_x & -\sin\alpha_x & 0 \\ 0 & \sin\alpha_x & \cos\alpha_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

mit $\alpha_x = \arccos \frac{(\mathbf{R}_y d)_y}{\|\mathbf{Rd}\|}$.

Korrektur der Abstände von Hüfte zu Knie bzw. Knie zu Knöchel

Nach der Berechnung der Transformationsmatrizen **S** und **R** für Skalierung und Rotation werden die drei anatomischen Bereiche auf der y-Achse angeordnet. Dazu werden die Koordinaten der Translation

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & 0 & -(\mathbf{RSb})_x \\ 0 & 1 & 0 & m_y \\ 0 & 0 & 1 & -(\mathbf{RSb})_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

unterzogen. Dabei ist $(\mathbf{RSb})_x$ die x-Koordinate und $(\mathbf{RSb})_z$ die z-Koordinate des skalierten und rotierten Stabanfangspunktes **b**. Die Translation bewirkt, dass der Stab in der y-Achse zu liegen kommt (vergleiche Abbildung 5.12). m_y stellt eine sequenzspezifische Translation dar, die den rekonstruierten Markerpunkt der jeweiligen Szene an die Position -35cm, 0cm oder 47cm auf der y-Achse verschiebt. Dies sind genau die originalen Abstände der Markierungen auf dem Kalibrierungsstab.

Ausrichtung der Sequenzen an einer Ebene

In diesem letzten Schritt der Ausrichtung werden die rekonstruierten Szenen mit Hilfe der Diamanten an einer Ebene ausgerichtet. Dazu ist lediglich eine Rotation um die y-Achse notwendig, da alle anderen Freiheitsgrade in den vorherigen Abschnitten bereits eleminiert wurden.
Bei der Anfertigung der Aufnahmen liegen jeweils Ober- bzw. Unterseite der Kalibrierungsquader in einer Ebene, während die Seitenflächen mehr oder weniger beliebig ausgerichtet sein

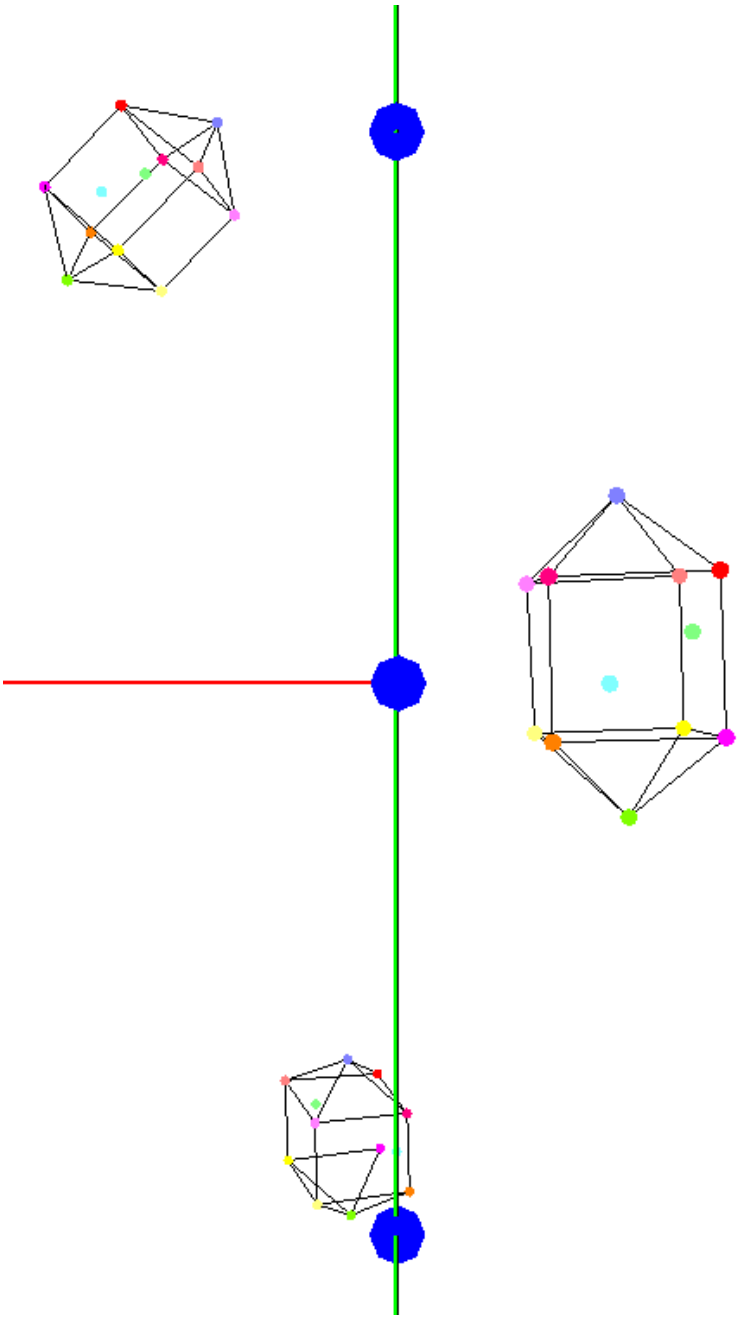


Abbildung 5.12: Die rekonstruierte Szenen nach der Translation entlang der y-Achse

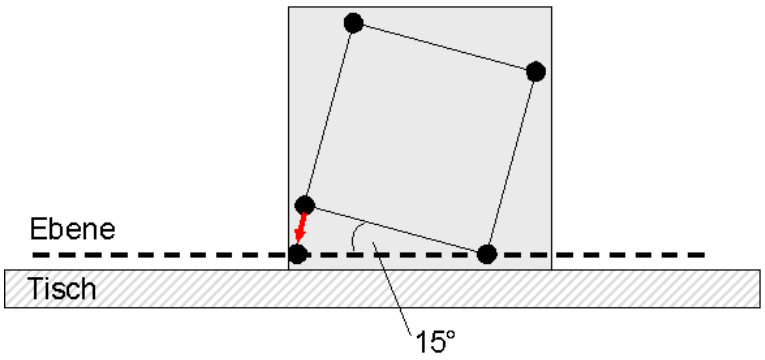


Abbildung 5.13: Würfelausrichtung: Der um 15° gedrehte Diamant wird in eine Ebene parallel zur Tischfläche gedreht.

können. Da der Diamant innerhalb des Quaders aus produktionstechnischen Gründen um 15° geneigt wurde und nicht bekannt ist, welche Seite auf dem Tisch liegt, müssen zunächst die entsprechenden Flächen berechnet werden. Von den vier quadratischen Seitenflächen des Würfels wird zunächst diejenige bestimmt, welche am nächsten zum rekonstruierten Stab liegt. Dazu wird der durchschnittliche Abstand der Eckpunkte der vier Flächen zum Stab berechnet. Anschließend wählt der Algorithmus vier zu einer angrenzenden Seitenfläche gehörenden Eckpunkte aus. Da für alle drei rekonstruierten Diamanten dieselbe angrenzende Fläche gewählt wird, erhält man so bei allen die Ober- oder bei allen die Unterseite des Kalibrierungsobjektes. Von den vier gewählten Punkten werden nun zwei so verschoben, dass alle vier in einer zur Quaderoberfläche parallelen Ebene liegen (siehe Abbildung 5.13).
Der Winkel zwischen den in die xz-Ebene projizierten Normalen dieser Oberfläche und der yz-Ebene (o.B.d.A. werden die entsprechenden Seitenflächen in die yz-Ebene gedreht) liefert den gesuchten Rotationswinkel um die y-Achse und damit die Ausrichtungsrotation $\mathbf{R}_a$.

Die gesamte Transformation jeder Sequenz ergibt sich zu

$$\mathbf{M}_{ges} = \mathbf{R}_a\mathbf{T}\mathbf{R}_{ges}\mathbf{S}$$

Das Ergebnis ist in Abbildung 5.14 zu sehen.

5.8 Fazit

Insgesamt erhält man mit dem Rekonstruktionsalgorithmus aus den drei eingegebenen Bildsequenzen einen kompletten Satz an Projektionsmatrizen und Kamerapositionen. Mit Hilfe dieser Informationen können beliebige, in mindestens zwei Aufnahmen markierte anatomische Punkte und Achsen dreidimensional rekonstruiert werden. Unter Verwendung der entwickelten Kalibrierungsobjekte Diamant und Stab ermöglicht es schließlich der ausgefeilte Ausrichtungsalgorithmus, die drei rekonstruierten anatomischen Bereiche zueinander in Beziehung zu setzen und somit die Abweichungen der Achsen festzustellen.

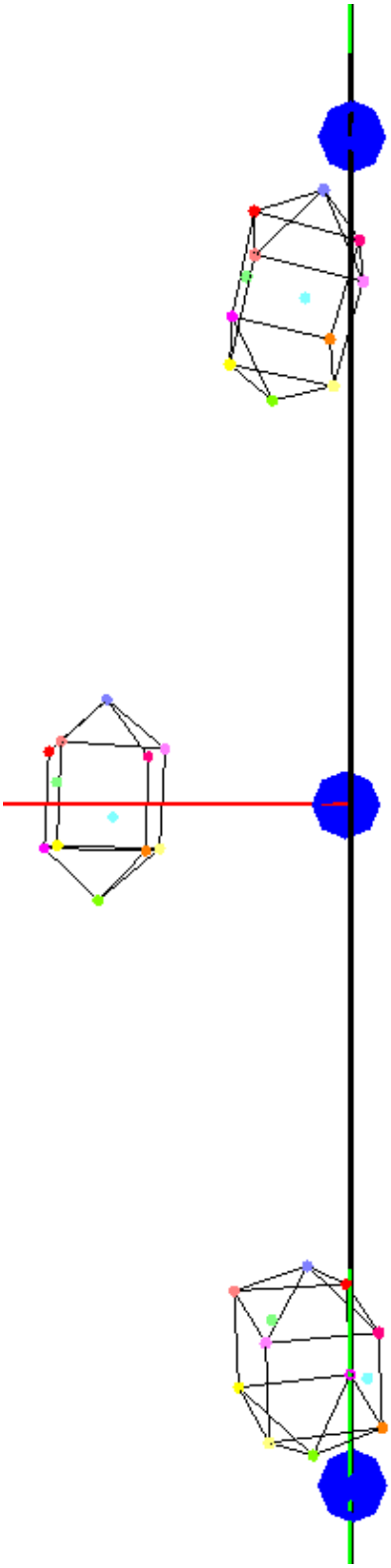


Abbildung 5.14: Ergebnis nach der Ausrichtung der Ebenen

Kapitel 6

Persistenz

Um Projekte sichen und laden zu können, mussten einige Klassen persistent gemacht werden. Als Persistenzmedium hat sich die Sprache XML [W3C04a] als optimale Lösung angeboten. Die Extensible Markup Language (XML) bietet ein textbasiertes Persistenzformat das eine menschenverständliche Lesbarkeit ermöglicht und zudem eine Austauschbarkeit mit anderen Anwendungen problemlos ermöglicht.

6.1 XML

In XML, als Textformat, werden Daten durch Angabe von sogenannten *Tags* als Typ definiert. Neben den *Tags* bietet XML die Möglichkeit Attribute als zusätzliche Dateninformation anzugeben. Dabei sind Attribute Paare aus Name und Wert, beispielsweise

```
<Clippingmask Rot="0" X="720" Y="576" >
```

wobei Clippingmask der entsprechende *Tag* und Rot, X, Y die entsprechenden Attribute sind. Dabei ist zu beachten, dass ein XML-Dokument syntaktisch korrekt sein muss, um mit den diversen XML-Dokument-Prozessoren, wie *DOM* [W3C04b] oder *SAX* [XML98] verarbeitet werden zu können. Valide ist ein Dokument dann, wenn alle eröffnenden *Tags* durch schließende *Tags* (`<tag>...</tag>`) beendet werden. Die Reihenfolge der eröffnenden *Tags* und schließenden *Tags* ist wichtig.

6.2 Qt DOM

Da der *SAX* Dokumenten-Parser ein ereignisbasiertes System, bei dem das gesamte Dokument sequentiell durchlaufen und jede gefundene Information durch einen Methodenaufruf an die zu verarbeitende Stelle propagiert wird, ist, hat sich die Wahl des *Qt DOM* Parsers angeboten. Das Document Object Model (*DOM*) ermöglicht einen Zugriff auf jede Stelle des XML-Dokumentes und direkten Zugriff auf jedes einzelne Attribut über die API-Funktion `QDomElement::attribute(“Attributname“,“Standardwert“)`. Zudem biete die „Qt DOM API“ Möglichkeiten um einen *DOM*-Baum aufzubauen und ihn dann durch einfachen Methodenaufruf in eine XML-Datei persistent zu machen.

6.3 GenuTEP-Persistenz-Dokument

Die GenuTEP-Persistenz-Datei, als XML-Dokument, baut einen Baum aus den persistenten Feldern, der zu persistierenden Klassen, auf. Das Hauptelement bildet das Project und ist als

```
<Project>
```

-Tag repräsentiert. Die direkten Kinder des Projects sind die Patientendaten, der Container für die Bilddaten, der Rekonstruktionscontainer, die Clippingmaske, die globalen Anwendungsoptionen und die intrinsischen Kameraparameter. Die Clippingmaske, wie auch alle anderen GenuBitMap-Objekte werden durch ein Run-Length-Encoding speichersparend mit gesichert. Der Bilddatencontainer gliedert sich in die drei Bereiche Hüfte, Knie und Knöchel auf und enthält in ihnen die Bildsequenzen der entsprechenden Daten. Eine solche Sequenz gleidert sich in die Rekonstruktionsinformation (GeoData) und mehrere GenuImages auf. Ein GenuImage enthält einerseits einen Verweis auf seine dewarpten Bilddaten im PNG-Format, andererseits einen Segmentierungscontainer für alle in diesem Bild erkannten Segmente.

```
<!DOCTYPE GenuTEP_Persistence>
<GenuTEP_Persistence>
 <Project>
  </Clippingmask>
  </Options>
  </Patient>
  <PictureDataSet>
   <ImageSequenceHip>
    <GenuImage>
     </SegmentationContainer>
    </GenuImage>
    </GeoData>
   </ImageSequenceHip>
   </ImageSequenceKnee>
   </ImageSequenceAnkle>
  </PictureDataSet>
  </ReconstructionContainer>
 </Project>
</GenuTEP_Persistence>
```

Kapitel 7

Werkzeuge

7.1 Qt

Qt, der Firma Trolltech [Tro03], ist eine Framework für die Anwendungsentwicklung unter C++. Neben unzähligen Werkzeugen und Utility-Klassen bietet Qt vor allem ein API für eine plattformunabhängige Implementation von graphischen Benutzeroberflächen. Qt ist für die nichtkomerzielle Nutzung unter Linux kostenlos erhältlich, für Windows und andere Betriebssysteme, sowie komerzielle Nutzung sind Lizenzen zu erwerben.

7.1.1 Aufbau von Qt

Ein komfortabler Editor (Qt-Designer, Abbildung 7.1) ermöglicht das visuelle Erstellen der einzelnen Fenster aus einzelnen Fensterelementen, so genannten Widgets. Das Layout wird in XML Dateien gespeichert. Das mitgelieferte Programm `uic` generiert daraus C++ Dateien, die komplette Fensterbeschreibungen als Klassen enthalten und normal editierbar sind. Dies wird durch `qmake` gewährleistet, welches das `Makefile`, das für die Steuerung der Kompilation und des Linken der Applikation zuständig ist, konsistent hält. Dazu werden die zu einem Programm gehörenden Quelldateien in einer Qt - eigenen Projekt-Datei (`.pro`) gespeichert, aus der `qmake` die nötigen Informationen für das `Makefile` erhält.
Das Toolkit ist vollständig objektorientiert aufgebaut, als abstrakte Basisklasse für alle Objekte von Qt dient `QObject` (Abbildung 7.2).

7.2 Open GL

OpenGL 1.2 [Ope04] ist die Basis für alle berechneten grafischen Darstellungen. Zum Einen wird diese Bibliothek zum Zeichnen von geometrischen Primitiven zur Interaktion genutzt. Im entwickelten Programm wurden Geometrien zur Visualisierung des Achsenrekonstruktion genutzt.

7.3 Newmat

Die Matrizenbibliothek Newmat [Dav04] in der Version 10 bietet Datenstrukturen und Operationen, zur effizienten Durchführung von Matrizenrechnungen. Einige der im Programm genutzten

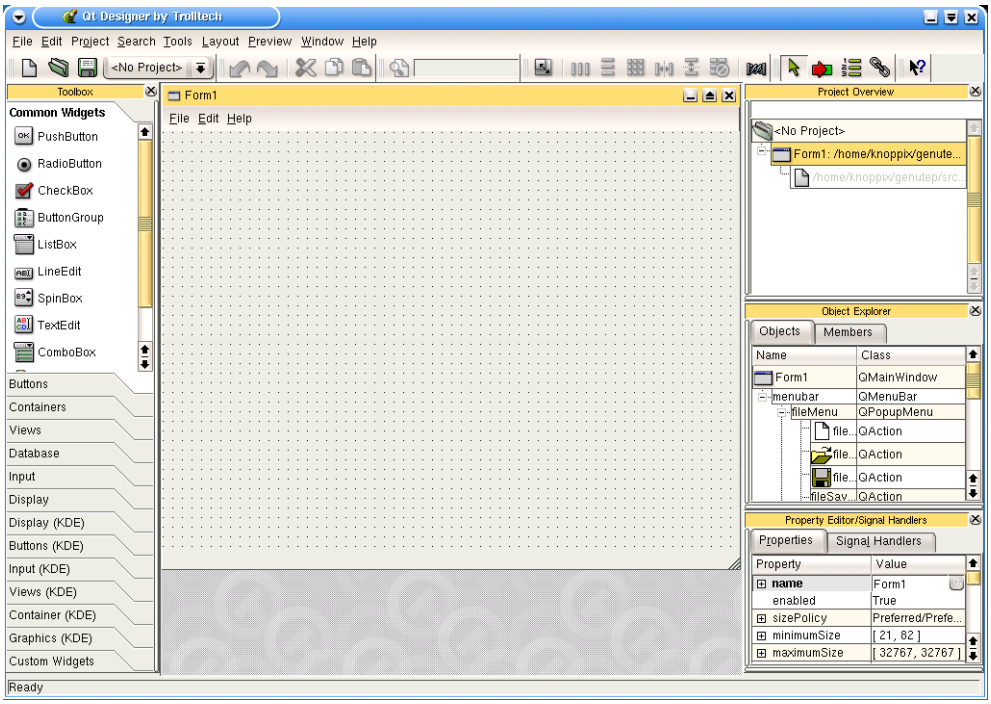


Abbildung 7.1: Qt Designer

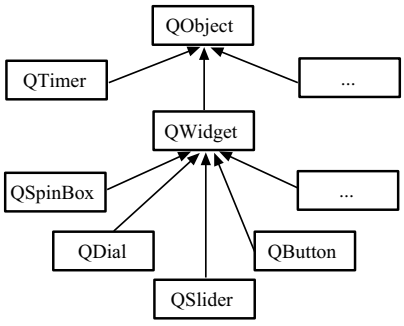


Abbildung 7.2: Objekthierarchie von Qt (Ausschnitt)

Standardklassen erben von Datenstrukturen der Bibliothek. Die genutzten Operationen sind u.a. die einfachen Matrizenmanipulation wie Multiplikationen, Additionen und Invertierungen.

7.4 Qhull

Qhull stellt Funktionen zur Verfügung, um z.B. konvexe Hüllen, Delaunay Triangulierungen oder Voronoi Diagramme zu berechnen. Hier wird es zum Aufbau der Clippingmaske verwendet, welche über die konvexe Hülle der Punkte des Kalibrierungsgitters definiert wird.

7.5 Together

Mit Together [Bor04] wurden Klassendiagramme und Sequenzdiagramme erstellt. Together erstellt aus dem Klassendiagramm Klassenrümpfe, in denen sämtliche Assoziationen, Vererbungen und Aggregationen in nutzbaren Code umgesetzt sind. Together bietet ebenfalls einen *Reverse-Engineering*-Modus an, bei dem aus bestehenden Code entsprechende Diagramme erzeugt werden könne.

7.6 Visio

Da Together als echtes OOD-Werkzeug anfangs nicht zur Verfügug stand wurde Visio [Mic04] zur Erzeugung von State-Charts benutzt. Visio ist ein Modellierungswerkzeug, dass neben UML duzende weitere Sprachen unterstützt.

7.7 CVS

CVS (Concurrent Versions System) [Ced03] ist ein System zur Versionsverwaltung von Dateien. Die Software ist unter der GPL für die meisten Betriebssysteme frei verfügbar. Das Konzept sieht einen geschützten Bereich, das so genannte Repository, vor, in dem alle Dateien inklusive deren Änderungen gespeichert werden. Dies kann entweder ein lokales Verzeichnis sein, oder über einen CVS Server gekapselt werden.

Für unser Projekt wird die Versionskontrolle mit Hilfe eines CVS Servers in Anspruch genommen, daher wird im Folgenden auf dieses Einsatzszenario eingegangen. CVS ist ein Kommandozeilenprogramm, die angegebenen Befehle müssen auf der Konsole eingegeben werden. Der Benutzer muss sich in einem Verzeichnis befinden, in dem er Schreib- und Leserechte besitzt.

7.8 Kdevelop

Mit Kdevelop [KDe03] in der Version 3 ist eine leistungsfähige Integrierte Entwicklungsumgebung für Linux verfügbar. Das Programm ist unter der GNU Lizenz verfügbar, d.h. es darf frei weitergegeben und benutzt werden.

In Abbildung 7.3 sieht man die Oberfläche von Kdevelop. Wichtige Befehle aus der Menüleiste sind über die Toolbar-Leiste darunter erreichbar. Diese ist den persönlichen Vorstellungen anpassbar, d.h. es kann individuell eingestellt werden, welche Befehle mit welchem Icon sichtbar

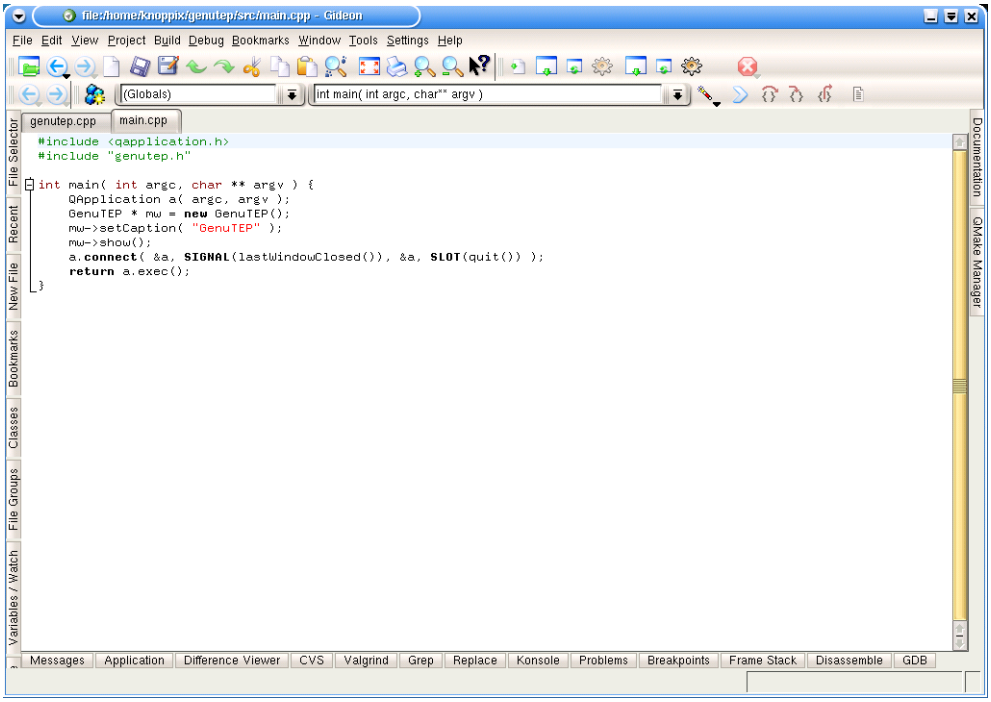


Abbildung 7.3: KDevelop

sein sollen, ebenso können Tastenkürzel frei definiert werden.

Kdevelop bietet eine von anderen intergrierten Entwicklungsumgebungen bekannte Projektverwaltung, fällt aber durch seine Vielseitigkeit aus dem Rahmen. Insbesondere ist die Programmiersprache nicht vorgegeben. Standardmäßig können bereits Projekte für C++, C, Java, Perl und andere Sprachen angelegt werden. Weitere Sprachen können über ein Plugin/ Template Konzept integriert werden.

7.9 Valgrind

Valgrind [KDE04] ist ein Open-Source Memory-Debugger-Tool unter dem Label der KDE-Organisation. Die Hauptaufgabe liegt in der Lokalisation von Speicherlöchern. Valgrind führt eine zu überprüfende Anwendung unter einem x86-Linux-Sytem in einem Kontrollmodus aus und Überwacht alle Speicherzugriffe. Die Allokation und Freigabe von Speicher wird dabei beobachtet und im Fehlerfall gemeldet. Valgrind ist für C und C++ Programme geschrieben worden, arbeitet aber auch mit in jeder anderen Sprache geschriebenen Programmen.

7.10 Latex

Mit LaTeX 2_ε wurden sämtliche Dokumente erstellt, die während der Projektphase zu erstellen waren. Zum einen wurden sowohl die wöchentlichen Sitzungsprotokolle als auch die umfangreicheren Berichte, wie Pflichtenheft, Seminarband und Zwischenbericht mit dem Textsatzsystem

verarbeitet.
Latex bietet umfangreiche Pakete zur Erstellung wissenschaftlicher Dokumente. Besonders hervorzuheben ist hierbei beispielsweise die Formelgebung sowie die Möglichkeit einzelne Dokumente in separaten Dateien abzuspeichern, um sie letztendlich in einem großen Dokument wie diesem Endbericht zusammenzufügen.

7.11 TWiki

Das TWiki [TWi04] ist ein web-basierte Kollaborationsplattform. Wikis, auch WikiWikis und WikiWebs, sind im World Wide Web verfügbare Seitensammlungen, die von den Benutzern nicht nur gelesen, sondern auch online geändert werden können. Sie sind damit offene Content Management Systeme. Der Name stammt von wikiwiki, dem hawaiianischen Wort für „schnell“. Wie bei Hypertexten üblich, sind die einzelnen Seiten und Artikel eines Wikis durch Querverweise (Links) miteinander verbunden. Die Seiten lassen sich jedoch sofort am Bildschirm ändern. Dazu gibt es in der Regel eine Bearbeitungsfunktion, die ein Eingabefenster öffnet, in dem der Text des Artikels bearbeitet werden kann. TWiki bietet zudem eine gegenüber HTML vereinfachte Syntax, um Formatierungen wie Aufzählungen, Nummerierungen, Schrifttypauswahlen, etc. zu vereinfachen.
Die PG434 hat ein TWiki-System auf dem Webservers des Lehrstuhls VII aufsetzen lassen. Es wurden Kommentare, TODOs und Projektentscheidungen dort eingepflegt.

Kapitel 8

Fazit

8.1 Die Projektarbeit

Zum Verlauf der Projektarbeit in den vergangenen zwei Semestern ist zunächst einmal folgendes anzumerken. Trotz der sehr zeitintensiven Arbeit am Projekt ist der Spaß an der Arbeit nicht zu kurz gekommen, sondern trug vielmehr intensiv zur Motivation bei. Dies lag zum einen an der sehr interessanten Aufgabenstellung und dem Willen etwas zu programmieren für das es eine reale Anforderung gab. Zum anderen an dem positiven Arbeitsklima innerhalb der Projektgruppe, das dazu beitrug, dass die Motivation meist gut war. Problematisch hingegen jedoch war die sehr hohe Einarbeitungszeit aufgrund der Komplexität des Themas und der Tatsache dass einige Teilbereiche doch sehr kompliziert zu verstehen waren. Aufgrund dessen mussten in vielen Bereichen diverse Verfahren getestet werden, bis sich ein Lösungsansatz als der Beste herausstellte.

Da sich die Projektgruppe dazu entschlossen hatte mit dem Prototypenmodell zu arbeiten, wurden im ersten Semester für die einzelnen Module die entsprechenden Prototypen entworfen und implementiert. Dies hatte den Vorteil, dass sich die Untergruppen in ihr Themengebiet einarbeiten und dabei wichtige Grundlagen erlernen konnten.

Im zweiten Semester konzentrierte sich die Projektgruppe darauf, die Schnittstellenspezifikation sowie die graphische Benutzerschnittstelle zu entwerfen und die einzelnen Prototypen im finalen Gesamtsystem zusammenzuführen. Die Weiterentwicklung der Module verlief parallel zu diesem Entwicklungsschritt bis zuletzt die Funktionalität des Programmes getestet wurde.
Abschließend bleibt festzuhalten, dass die Zielvorgaben aus dem Pflichtenheft realisiert wurden.

8.2 Danksagung

Die Mitglieder der Projektgruppe bedanken sich bei Herrn PD Dr. med. Klaus Schmidt (Kath. Krankenhaus Kirchlinde Dortmund-West) und Herrn PD Dr. med. Gebhard Schmid (Johanna-Etienne-Krankenhaus Neuss) sowie dem freundlichen Pflegepersonal für die konstruktive Zusammenarbeit. Ebenso danken wir dem Lehrstuhl 7 (Graphische Systeme) des Fachbereiches Informatik der Universität Dortmund für die Bereitstellung der Räumlichkeiten und der technischen Ausstattung. Einen besonderen Dank richten wir an Prof. Dr. Dietrich Wegener vom Lehrstuhl für Experimentelle Physik V der Universität Dortmund und Herrn Domke, Mechanische Werkstatt, für die Konzeption und Fertigung des Kalibrierobjektes. Außerdem bedanken wir

uns bei Martin Wawro und Frank Weichert für die gute Unterstützung.

Kapitel 9

Handbuch

9.1 Vorwort

Als Folge von Arthroseerkrankungen o.ä. muss oftmals das menschliche Kniegelenk komplett durch ein künstliches Kniegelenk ersetzt werden. Die totale Endoprothese für das menschliche Kniegelenk (kurz: GenuTEP) gehört zu den am häufigsten eingesetzten Prothesen in Deutschland. Auffallend ist, dass die Prothesen einiger Patienten größere Verschleißerscheinungen aufweisen als Andere. Es stellt sich die Frage, ob der Einbauwinkel der Prothese die Lebensdauer beeinträchtigt. Bei einer Knieoperation werden die Prothesenachsen oft nur näherungsweise an die Achsen des Ober- und Unterschenkelknochens ausgerichtet. Es wird vermutet, dass bei einem schlechten Einbauwinkel die Prothesen Belastungen unterliegen, die einen höheren Verschleiß der Prothese und des Knochens zur Folge haben. Wenn beim Eingriff die Prothesenachsen an den anatomischen Achsen der Knochen ausgerichtet werden, könnte die Lebensdauer der Prothesen verlängert werden.

9.1.1 Was ist GenuTEP

Das Programm “GenuTEP“ ermöglicht die postoperative Rekonstruktion der Achsen der implantierten Prothese und die visuelle Gegenüberstellung derselben in Bezug zu den anatomischen Achsen des menschlichen Ober- und Unterschenkel. Mit diesen Informationen sollen in der medizinischen Forschung Erkenntnisse gesammelt werden, um so die Lebensdauer von Prothesenimplantaten verlängern zu können.

9.1.2 Programmablauf

Nach dem Starten des Programms kann zunächst ein neues Projekt angelegt oder ein bereits vorhandenes Projekt geöffnet werden (siehe Abbildung 9.2). Beim Anlegen eines neuen Projektes werden die gewünschten C-Bogen-Sequenzen von Knie, Hüfte und Knöchel importiert. Der Benutzer wird dann durch den Projekt-Anlege-Manager geführt, in dem er bestimmte Eingaben (Projektname, Patientenname, . . .) tätigen muss. Es kann notwendig sein, dass Bilder geclippt und gedreht werden müssen (siehe Abschnitte 9.3.1.4 und 9.3.1.5). Es besteht nun die Möglichkeit aus den aufgenommenen Bildern bestimmte Bilder auszuwählen, die wichtig und geeignet erscheinen (siehe Abschnitt 9.3.3). Anschließend muss in allen drei Sequenzen (Hüfte, Knie, Knöchel) der Würfel, die Prothese (nur Knie) und der Stab segmentiert werden (siehe Abschnitt 9.3.4). Die Segmentierung benötigt je nach Bildqualität manuelle Unterstützung. Der

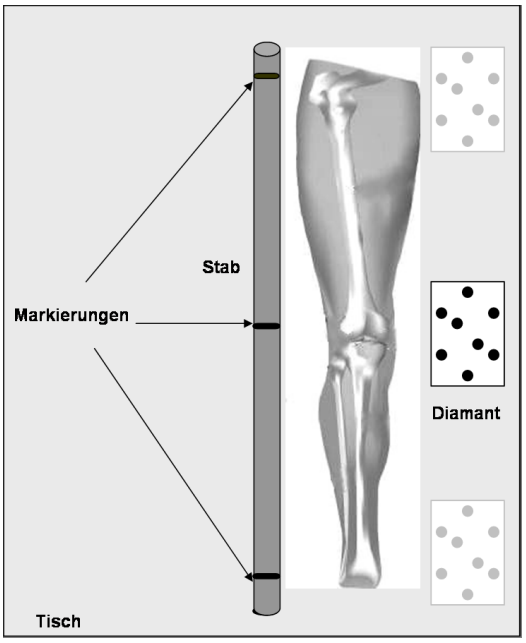


Abbildung 9.1: Anordnung der Kalibrierobjekte

anschließende Schritt, die Rekonstruktion der Achsen, läuft vollautomatisch ab (siehe Abschnitt 9.3.5). Die Achsenrekonstruktion kann dann als 2D- oder 3D-Ansicht betrachtet werden (siehe Abschnitte 9.3.7 und 9.3.6).

9.2 Installation

9.2.1 Hardwarevoraussetzungen

Das Programm läuft auf einem PC mit folgenden Hardware-Mindestanforderungen:

1. Intel Pentium 4 Dual (2.0 GHz)
2. 1 GB RAM
3. Grafikkarte: nVidia GeForce 4 (256 MB)

Erzeugt der C-Bogen digitalisierte Daten im DICOM-Format, so können diese direkt verwendet werden. Ansonsten kann zur Digitalisierung der Daten aus einem videokompatiblen C-Bogen eine Framegrabber-Karte eingesetzt werden.

Zur Entzerrung der Bilder (Dewarping) wird ein planares Kalibrierungsgitter benutzt. Bei der erstmaligen Verwendung eines C-Bogens in Verbindung mit dem genutep-Programm muss zunächst der C-Bogen kalibriert werden (siehe Abschnitt 9.4).
Als weitere Kalibrierinstrumente werden der Kalibrierdiamant und der Kalibrierstab benötigt. Diese müssen auf jeder Fluroskopieaufnahme erkennbar sein und deren Position darf zwischen den einzelnen Aufnahmen nicht verändert werden.

In Abbildung 9.1 ist die Anordnung der Kalibrierobjekte dargestellt. Der Kalibrierstab wird direkt neben das aufzunehmende Bein gelegt, so dass die drei Markierungen in Höhe der Hüfte, des Knies und des Knöchels liegen. Zunächst sollte der Diamant neben den Knöchel gelegt werden. Der C-Bogen sollte so positioniert werden, dass Stab, Knöchel und Diamant auf dem Bild sichtbar sind. In dieser Einstellung werden nun aus unterschiedlichen Winkeln mehrere (mindestens drei) Aufnahmen gemacht. Die Position des C-Bogens darf zwischen den einzelnen Aufnahmen nicht verändert werden. Für die anderen beiden anatomischen Bereiche (Hüfte und Knie) wird analog vorgegangen. Die Position des Stabes darf zwischen den Aufnahmen der verschiedenen Bereiche nicht verändert werden! Außerdem darf sich der Patient während der gesamten Aufnahmeprozedur nicht bewegen.

9.2.2 Softwarevoraussetzungen

Das Programm läuft unter Linux oder Windows.
Zur grafischen Ausgabe wird unter Linux X-Windows (X11R6) benötigt, wobei auf die Qt-Bibliothek (Version 3.x.x) und auf OpenGL (Version 1.3) zurückgegriffen wird.
Unter Linux ist die glibc ab Version 2 als zu verwendende C-Bibliothek vorausgesetzt.
MS-Windows-basierte Betriebssysteme werden in den Versionen Windows 2000 Professional und Windows XP (Pro, Home) unterstützt.

9.3 Bedienung des Programms

In diesem Abschnitt erfolgt eine genaue Beschreibung der einzelnen Programmschritte. Zunächst wird beschrieben, auf welche Weise ein Projekt neu angelegt wird. Anschließend wird ein Überblick über die Funktionen der Toolbar und der Menüleiste des Hauptfensters gegeben. Abschließend werden alle Dialoge, in der Reihenfolge, in der sie beim Durchlauf durch das Pogramm auftreten, erklärt. Das Kalibrieren der Kamera ist im Anhang (siehe Kapitel 9.4, Seite 141) beschrieben.

9.3.1 Anlegen eines neuen Projektes

Um ein neues Projekt anzulegen, kann im Projekt-Menü `Anlegen` gewählt werden oder der entsprechende Button der Toolbar angeklickt werden. Dann wird der gesamte Projekt-Anlege-Assistent durchlaufen. In diesem müssen verschiedene Daten und Verzeichnisse angegeben und ausgewählt werden. Die einzelnen Dialoge des Projekt-Anlege-Assistenten werden in der zu durchlaufenden Reihenfolge detailliert beschrieben. Zur Navigation zwischen den Dialogen stehen die drei Buttons `Next`, `Back` und `Cancel` zur Verfügung. Der `Next`-Button führt zum nächsten, der `Back`-Button zum vorherigen Dialog. Mit `Cancel` kann das Anlegen eines neuen Projektes abgebrochen werden. Der letzte Dialog enthält statt des `Next`-Buttons einen `Finish`-Button, mit dem das neue Projekt erzeugt, der Projekt-Anlege-Assistent beendet und das Hauptfenster (siehe Kapitel 9.3.2) geöffnet wird.

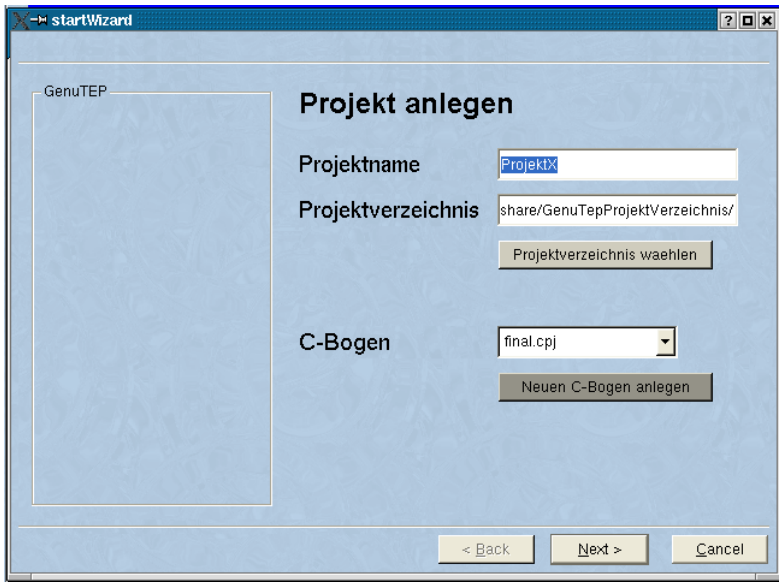


Abbildung 9.2: „Projekt anlegen“- Dialog

9.3.1.1 „Projekt anlegen“-Dialog

In diesem Dialog (siehe Abbildung 9.2) müssen der Projektname, das Projekt-Verzeichnis und der entsprechende C-Bogen ausgewählt bzw. angelegt werden.

Projektname
In diesem Feld muss der Projektname angegeben werden. Als Projektname können beispielsweise die Patientennummer, der Patientenname oder Kombinationen davon gewählt werden.

Projektverzeichnis
Angabe des Projektverzeichnisses, in dem die Projektdatei (.gtp) und die dewarpten Bilder gespeichert werden.

Projektverzeichnis wählen
Um den Verzeichnispfad nicht per Hand eingeben zu müssen, kann der entsprechende Pfad über diesen Button gewählt werden.

C-Bogen
Hier muss der C-Bogen ausgewählt werden, mit dem die zu bearbeitenden Bilder aufgenommen wurden.

Neuen C-Bogen anlegen
Falls der benötigte C-Bogen nicht in der Liste `C-Bogen` aufgeführt ist, muss ein neuer C-Bogen angelegt werden. Dies beinhaltet das Durchlaufen der Kalibrier- und Dewarping-Routine. Die Kalibrier-Routine ist in Kapitel 9.4.1 auf Seite 141 erklärt.

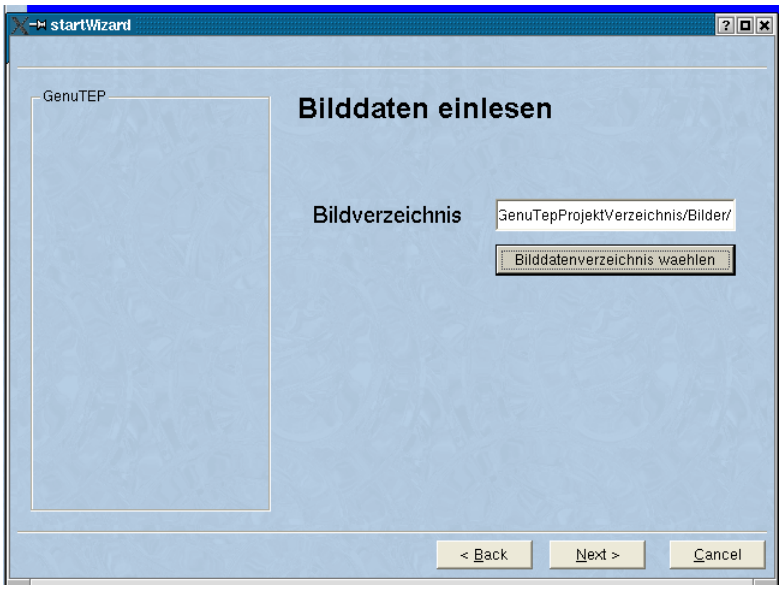


Abbildung 9.3: Bilddaten einlesen

9.3.1.2 „Bilddaten einlesen“-Dialog

Abbildung 9.3 zeigt den Dialog zum Einlesen der Bilddaten. Hier muss das Verzeichnis, in dem die entsprechenden Bilder für das vorher angelegte Projekt liegen, angegeben bzw. ausgewählt werden.

Bildverzeichnis
: Hier steht der Pfad des gewählten Bildverzeichnisses. Dies Verzeichnis muss drei Unterverzeichnisse „Hüfte“, „Knie“ und „Knöchel“ mit den entsprechenden Fluoroskopieaufnahmen enthalten.

Bildverzeichnis wählen
: Über diesen Button kann das gewünschte Verzeichnis ausgewählt werden. Hierzu öffnet sich der Standard-Dialog zum Auswählen von Verzeichnissen.

Next
: Die gewählten Bilder in dem Verzeichnis werden dewarped. Das Fortschreiten des Dewarpings ist an dem eingeblendeten Prozentbalken ablesbar.

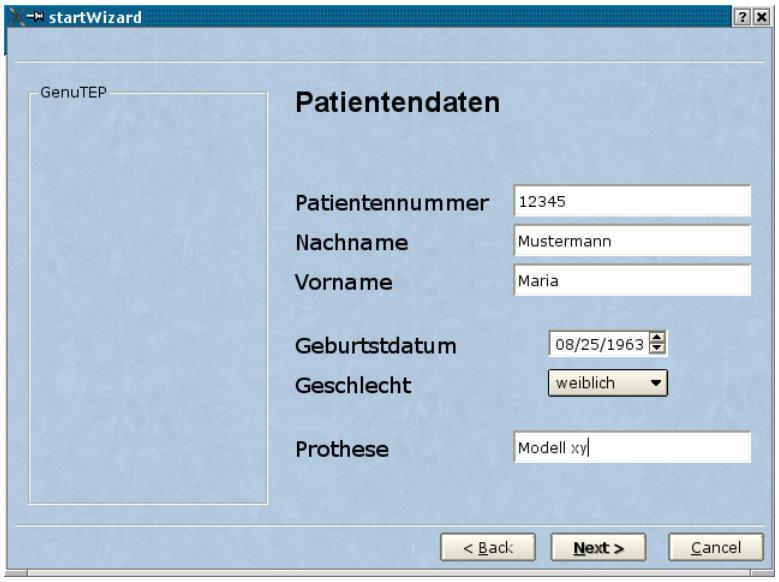


Abbildung 9.4: Eingabe der Patientendaten

9.3.1.3 „Patientendaten“-Dialog

Im „Patientendaten“-Dialog (siehe Abbildung 9.4) werden die Patientendaten automatisch angezeigt, falls die Bild-Daten im DICOM-Format vorliegen. Andernfalls müssen die Daten per Hand eingetragen werden.

Patientennummer
In diesem Textfeld wird die Patientennummer angezeigt oder muss eingetragen werden. Es sind alphanumerische Patientennummern möglich.

Nachname
Der Patientennachname wird angezeigt oder muss angegeben werden. Im Beispiel (siehe Abbildung 9.4) wurde hier „Mustermann“ eingetragen.

Vorname
Hier wird der Patientenvorname eingetragen. Beispielhaft wurde hier als Vorname „Maria“ angegeben.

Geburtsdatum
Das Geburtsdatum des Patienten wählen. Entweder mit Hilfe der kleinen Pfeile oder durch Eingabe per Tastatur möglich.

Geschlecht
Das Geschlecht des Patienten auswählen. Im Beispiel wurde eine Patientin gewählt.

Prothese
Den Namen der Prothese angeben. Sinnvoll wäre hier z.B. die Angabe des Herstellers und die genaue Modellbezeichnung.

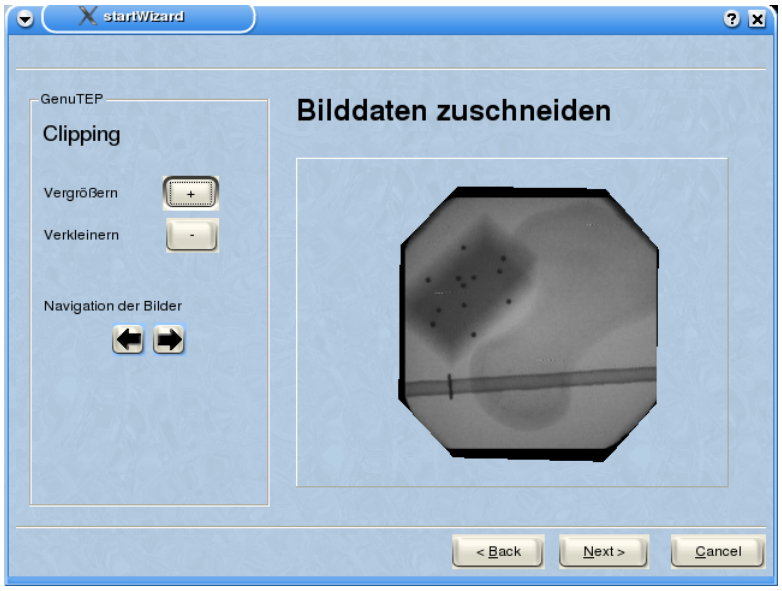
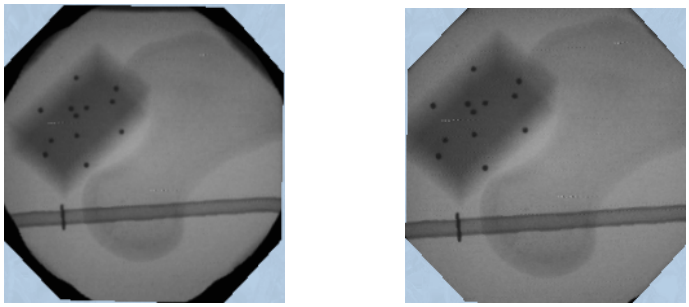


Abbildung 9.5: Bilddaten zuschneiden

9.3.1.4 „Bilddaten zuschneiden“-Dialog

Aufgrund des beschränkten Bildausschnittes des C-Bogens können die Aufnahmen teilweise einen störenden schwarzen Rand aufweisen. Da der schwarze Rand bei der Anwendung der Algorithmen zu Störungen führt, muss in diesem Dialog (siehe Abbildung 9.5) das Bild so zugeschnitten werden, dass nur noch die wichtigen Bildinformationen sichtbar sind (möglichst kein schwarzer Rand). In Abbildung 9.6 ist ein guter und ein schlechter Zuschnitt dargestellt. Falls der relevante Bildbereich in der ersten Aufnahme nicht gut genug erkennbar sein sollte, kann mit Hilfe der Buttons mit den Pfeilen (siehe Abbildung 9.5) ein anderes Bild gewählt werden.



(a) Schlecht (b) Gut

Abbildung 9.6: Beispiel für einen guten bzw. schlechten Zuschnitt der Bilder

Clipping

Mit der „+“-Taste kann der Clippingbereich vergrößert werden (allerdings nur bis zur Originalgröße des Bildes). Somit dient er zur Korrektur, falls zuviel Bildinformation abgeschnitten wurde. Mit der „-“-Taste wird der Clippingbereich schrittweise verkleinert bis die gewünschte Größe erreicht ist.

Navigation der Bilder

Mit den Pfeiltasten kann zwischen den Bildern der ausgewählten Bildsequenz gewechselt werden. Mit dem rechten Pfeil gelangt man zum nächsten Bild, mit dem linken Pfeil zum vorherigen Bild.

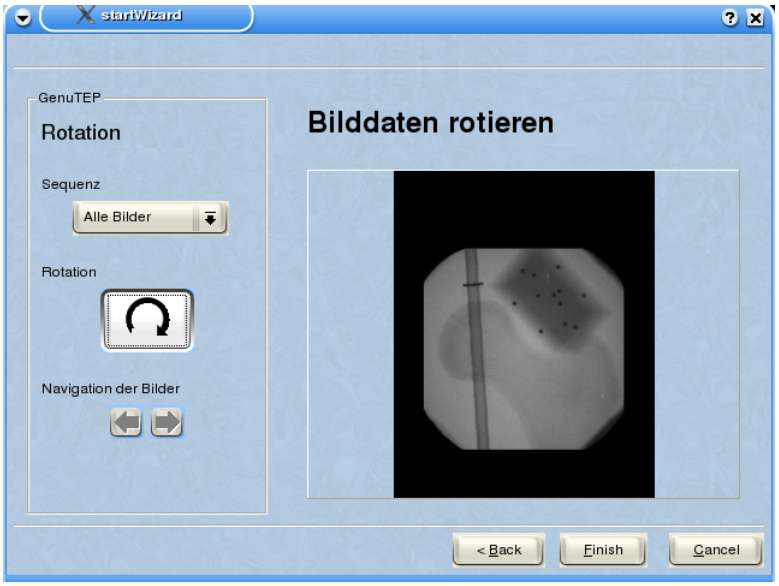


Abbildung 9.7: Bilddaten rotieren

9.3.1.5 „Bilddaten rotieren“-Dialog

Je nach Position des C-Bogens müssen die akquirierten Aufnahmen so gedreht werden, dass der Stab immer vertikal auf dem Bild zu sehen ist. Die senkrechte Lage des Stabes in allen drei Sequenzen ist zwingend erforderlich, damit die automatische Segmentierung erfolgreich durchgeführt werden kann. Die drei anatomischen Teilbereiche (Hüfte, Knie, Knöchel) können in der ComboBox ausgewählt werden (siehe Abbildung 9.7). Es müssen alle drei anatomischen Teilbereiche überprüft werden, da der C-Bogen bei den Aufnahmen der drei Teilsequenzen beliebig um den Patienten herum positioniert werden kann. Somit muss die Ausrichtung der Bilder der einzelnen Teilsequenzen nicht identisch sein. Durch Auswahl des Eintrags „Alle“ ist auch das gleichzeitige Rotieren aller drei Sequenzen möglich.

Rotation
: Mit diesem Button kann das Bild in 90°-Schritten gedreht werden. Durch mehrmaliges Betätigen der Taste kann das Bild in die richtige Position (s.o.) gebracht werden.

Navigation der Bilder
: Mit den Buttons mit den Pfeilen kann zwischen den Bildern der ausgewählten Bildsequenz gewechselt werden. Mit dem rechten Pfeil gelangt man zum nächsten Bild, mit dem linken Pfeil zum vorherigen Bild.

Sequenz
: In der ComboBox kann zwischen den einzelnen anatomischen Teilbereichen (Hüfte, Knie, Knöchel, alle Bilder) gewählt werden. Ein Bild der ausgewählten Sequenz wird angezeigt.

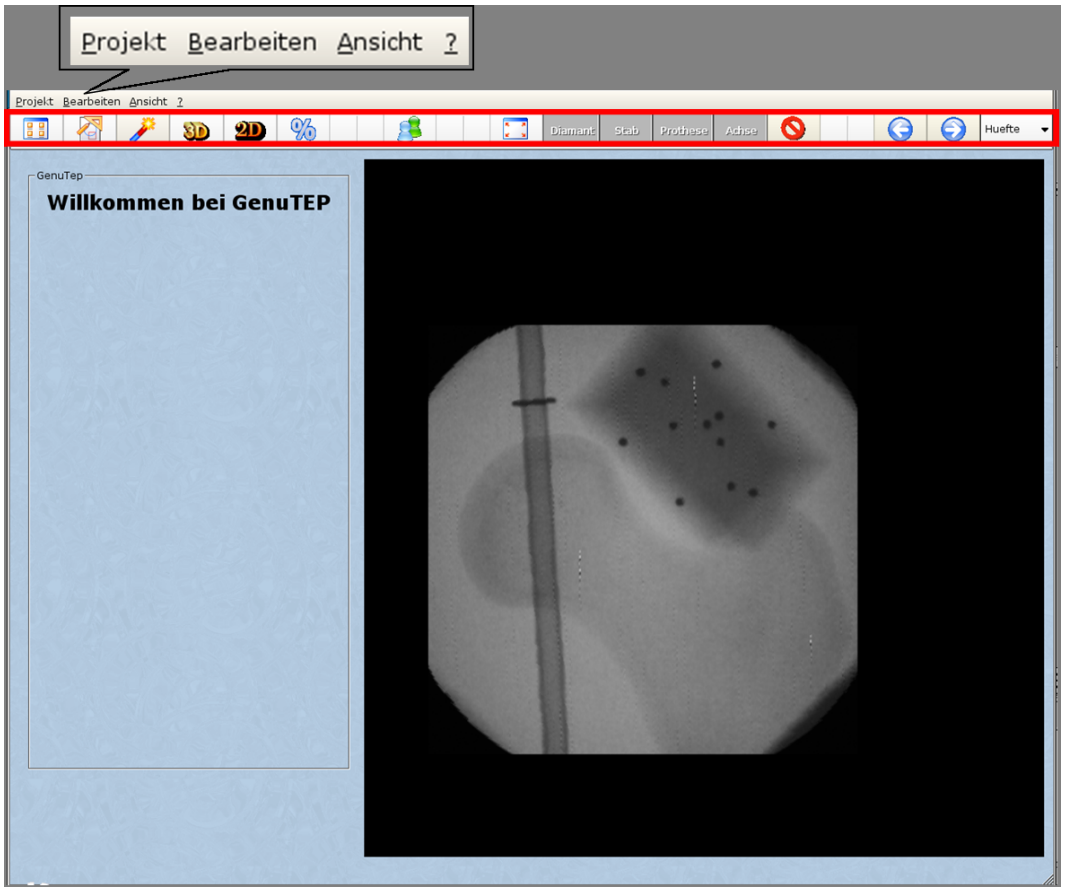


Abbildung 9.8: Hauptfenster

9.3.2 Das Hauptfenster

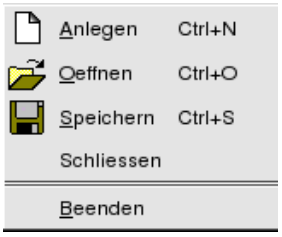
Die Komponenten des Hauptfensters (siehe Abbildung 9.8), die permanent zu sehen sind, bilden die Toolbar (rot umrandet) und das Hauptmenü (vergrößert dargestellt). Darüber hinaus ist links immer der Assistent eingeblendet, in dem Erklärungen zu den einzelnen Punkten sowie die entsprechenden Buttons für gerade auszuführende Aktion eingeblendet werden. Im rechten Bereich des Hauptfensters ist die 2D- bzw. 3D-Ansicht zu finden.

9.3.2.1 Menü

In diesem Abschnitt werden die einzelnen Menüpunkte (siehe Vergrößerung in Abbildung 9.8) beschrieben.

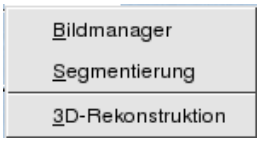
Projekt-Menü
Die Einträge haben folgende Bedeutung:

- **Anlegen:** Der Projekt-Anlege-Manager wird geöffnet
- **Öffnen:** Ein bereits angelegtes Projekt wird geöffnet
- **Speichern:** Das Projekt wird gespeichert
- **Schließen:** Das Projekt wird geschlossen
- **Beenden:** Das Programm wird geschlossen



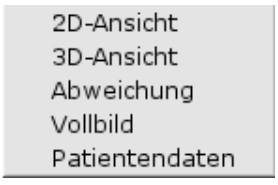
Bearbeiten-Menü
Die Einträge haben folgende Bedeutung:

- **Bildmanager:** Der Bildmanager wird geöffnet
- **Segmentierung:** Die Segmentierung wird gestartet
- **3D-Rekonstruktion:** Die 3D-Rekonstruktion wird gestartet



Ansicht-Menü
Die Einträge haben folgende Bedeutung:

- **2D-Ansicht:** Zeigt die Achsenrekonstruktion in der 2D-Ansicht.
- **3D-Ansicht:** Zeigt die Achsenrekonstruktion in der 3D-Ansicht.
- **Abweichung:** Zeigt die Abweichung an.
- **Vollbild:** Wechselt zwischen Vollbild und Fenstermodus des Programms.
- **Patientendaten:** Die Patientendaten werden angezeigt.



Hilfe-Menü
Die Einträge haben folgende Bedeutung:

- **Hilfe:** Das Hilfe-Fenster wird geöffnet.
- **About:** Das About-Fenster wird geöffnet.

Abbildung 9.9: Toolbar

9.3.2.2 Toolbar

Die gesamte Toolbar ist in Abbildung 9.9 dargestellt. Die einzelnen Buttons haben folgende Funktionen:

 Der Bildmanager (siehe Kapitel 9.3.3) wird geöffnet.

 Die Segmentierung wird gestartet (siehe Kapitel 9.3.4).

 Die 3D-Rekonstruktion wird gestartet.

 Zeigt die Achsenrekonstruktion in der 2D-Ansicht.

 Zeigt die Achsenrekonstruktion in der 3D-Ansicht.

 Die Abweichung wird angezeigt.

 Die Patientendaten werden angezeigt.

 Das aktuelle Bild wird im Vollbild Modus dargestellt.

 Das aktuelle Bild wird aus der Sequenz entfernt.

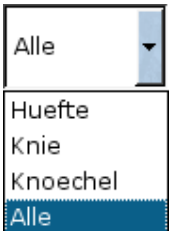
 Das vorherige Bild der Sequenz wird angezeigt.

 Das nächste Bild der Sequenz wird angezeigt.

Abbildung 9.10: Ausschnitt der Toolbar zur Ansicht der segmentierten Objekte

Segmentierungs-Ansicht
Mit den in Abbildung 9.10 zu sehenden Buttons können die einzelnen Segmentierungen der Objekte (Diamant, Prothese, Stab) ein- und ausgeschaltet werden. Falls die Punkte zur Ermittlung der Prothesenachsen schon eingezeichnet wurden, kann mit dem `Prothese`-Button auch die Anzeige der Prothesenachsen aktiviert bzw. deaktiviert werden. Die schon markierten Punkte zur Ermittlung der Ober- und Unterschenkelachse können mit dem `Achsen`-Button angezeigt werden. In der 3D-Ansicht werden mit dem `Prothese`-Button und dem `Achsen`-Button die rekonstruierten Prothesen- bzw. Ober-/Unterschenkel-Achsen ein- und ausgeblendet. In Abbildung 9.10 ist die Segmentierung der Prothese ausgeblendet während die Segmentierung der restlichen Objekte sichtbar ist.

Bereichsauswahl
Mit der Combobox rechts außen in der Toolbar kann man zwischen der Anzeige der verschiedenen anatomischen Bereiche (Knie, Hüfte, Knöchel) der Sequenz wählen. Wählt man z.B. `Knie` so werden nur Bilder der Knie-Sequenz angezeigt, segmentiert und 3D-rekonstruiert. Wählt man hingegen `Alle`, so werden alle drei Teilsequenzen angezeigt, segmentiert und 3D-rekonstruiert.



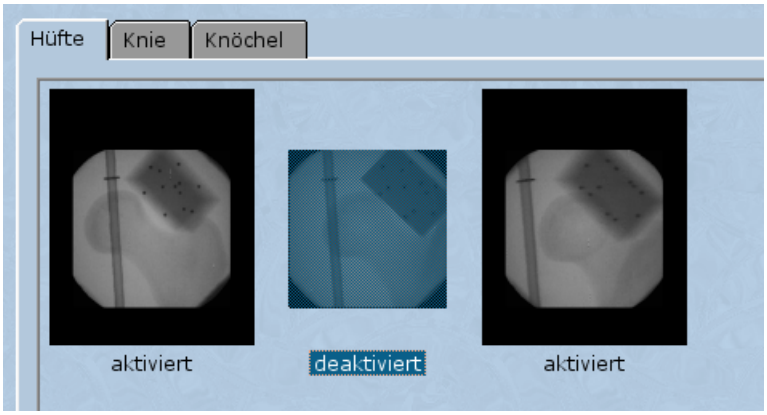


Abbildung 9.11: Bildmanager

9.3.3 Bildmanager

Im Bildmanager (siehe Abbildung 9.11) hat der Benutzer eine Übersicht über sämtliche Bilder der Sequenz. Die Bilder werden nach Bereichen (Hüfte, Knie, Knöchel) getrennt angezeigt. Es besteht die Möglichkeit Bilder, die von schlechter Qualität oder ungeeignet sind, zu deaktivieren. Darüber hinaus kann ein aktiviertes Bild mit Doppelklick ausgewählt werden und dieses wird dann im Hauptfenster angezeigt. Die Steuerung des Bildmanagers funktioniert folgendermaßen:

1. **Deaktivieren:**
   Mit der rechten Maustaste das Bild anklicken. Das deaktivierte Bild wird graphisch markiert und bei der Segmentierung und der Rekonstruktion nicht verwendet.

2. **Aktivieren:**
   Mit der linken Maustaste das Bild anklicken. Die Markierung wird aufgehoben und das Bild wird wieder in die Sequenz zum Segmentieren und Rekonstruieren aufgenommen.

3. **Bildauswahl:**
   Doppelklick mit der linken Maustaste auf ein aktiviertes Bild. Das ausgewählte Bild wird als aktuelles Bild in die 2D-Ansicht übernommen und der Bildmanager wird geschlossen.

4. **Beenden:**
   Mit der ESC-Taste kann der Bildmanager geschlossen werden. Dabei wird das vorherige aktuelle Bild beibehalten.

Hinweis:
Pro Sequenz müssen mindestens drei Bilder aktiviert sein. Falls nur noch drei Bilder einer Sequenz aktiviert sind, ist es nicht mehr möglich eines von diesen zu deaktivieren.

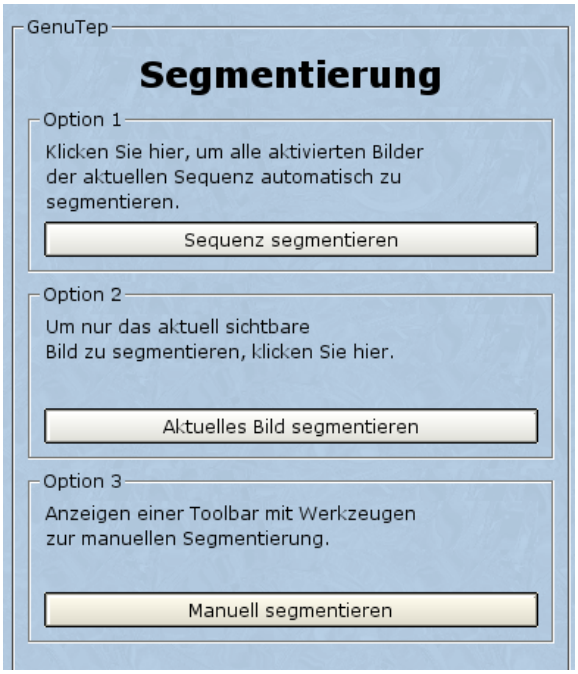


Abbildung 9.12: Segmentierung-Übersicht

9.3.4 Segmentierung

Wenn ein Projekt korrekt angelegt wurde, kann die Segmentierung gestartet werden. Ein vollständig segmentiertes Bild ist in Abbildung 9.13 zu sehen, Abbildung 9.14 zeigt eine schlechte Segmentierung. Ein Bild ist gut segmentiert, wenn die Segmentgrenzen mit der zu segmentierenden Struktur identisch sind, der Diamant als Diamant erkennbar ist und die Grenzen des Stabes sowie die Markierungen korrekt erkannt wurden. Wird die Segmentierung gestartet, gelangt man zu dem in Abbildung 9.12 dargestellten Assistenten. Hier besteht die Möglichkeit zwischen drei Optionen zu wählen.

1. **Automatisch segmentieren**
   Startet den Assistenten für die automatische Segmentierung für die Bilder der angegebenen Sequenz in der Toolbar (siehe Kapitel 9.3.2.2).

2. **Aktuelles Bild segmentieren**
   Nur das aktuell angezeigte Bild wird mit diesem Assistenten automatisch segmentiert.

3. **Manuell segmentieren**
   Jedes Bild und jedes zu segmentierende Objekt kann per Hand segmentiert werden.

Durch Anklicken des Segmentierungsbuttons (siehe Abschnitt 9.3.2.2, Seite 118) oder durch Auswahl des `Bearbeiten`-Menüeintrages `Segmentierung` gelangt man wieder zu dem in Abbildung 9.12 zu sehenden Assistenten.
Es empfiehlt sich zunächst die automatische Segmentierung durchlaufen zu lassen und anschließend, falls nötig, manuelle Korrekturen durchzuführen.

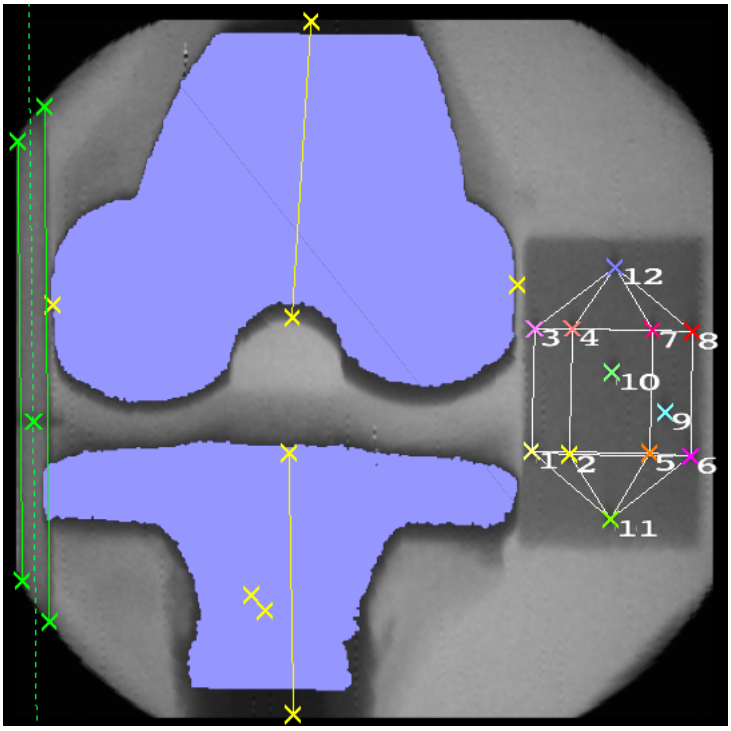


Abbildung 9.13: Korrekte Segmentierung einer Knie-Aufnahme

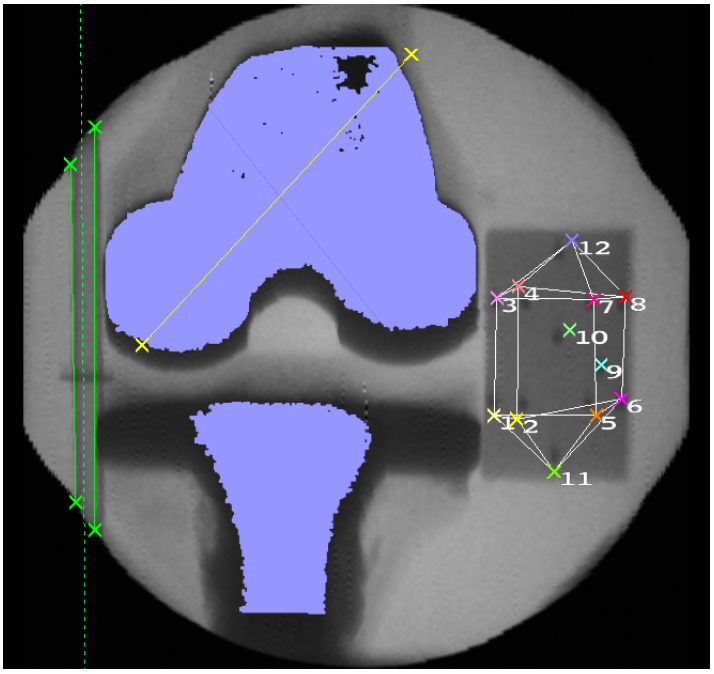


Abbildung 9.14: Schlechte Segmentierung einer Knie-Aufnahme

(a) Automatische Segmentierung

(b) Segmentierung des aktuellen Bildes

Abbildung 9.15: Automatische Segmentierung

9.3.4.1 Automatische Segmentierung

Der „Automatische Segmentierungs“-Assistent ist in Abbildung 9.15 zu sehen. Der Dialog hat folgende Funktionen:

Auswahl-Fenster
In dem `Auswahl`-Fenster können die zu segmentierenden Objekte ausgewählt werden. Ist eine Checkbox für einen Gegenstand mit einem Haken versehen, so wird dieser automatisch segmentiert.

Vorschau-Button
Mit dem `Vorschau`-Button wird das erste Bild segmentiert. So kann nachvollzogen werden, ob die automatische Segmentierung Erfolg versprechend ist, ohne die gesamte Segmentierung durchlaufen zu lassen.

Start-Button
Mit dem `Start`-Button wird die Segmentierung gestartet. Es ist bei jeder automatischen Segmentierung erforderlich, dass jedes Bild auf korrekte Segmentierung überprüft wird. Je nach Bildqualität muss die Segmentierung der Bilder noch manuell korrigiert werden (siehe Kapitel 9.3.4.3).

9.3.4.2 Aktuelles Bild segmentieren

In Abbildung 9.15 ist der Assistent für die Segmentierung des aktuellen Bildes zu sehen. So wie bei der automatischen Segmentierung kann ausgewählt werden, welche Objekte segmentiert werden sollen.
Mit `Start` wird die automatische Segmentierung für das momentan angezeigte Bild gestartet.

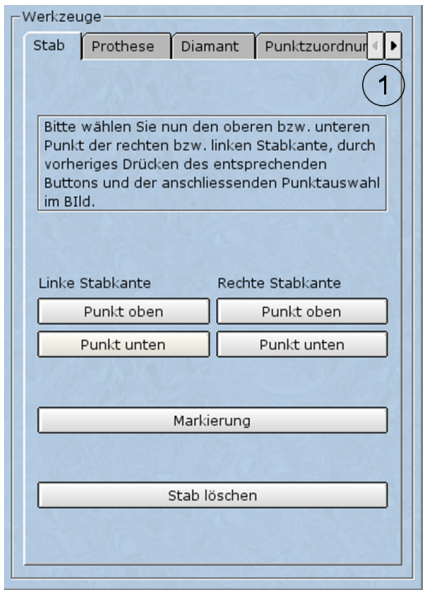


Abbildung 9.16: Manuelle Segmentierung des Stabes

9.3.4.3 Manuell segmentieren

Es ist möglich, dass die Ergebnisse der automatischen Segmentierung Korrekturen benötigen. Dies hängt von der Qualität der Bilder ab. Daher besteht die Möglichkeit die Objekte manuell zu segmentieren. Die Segmentierungs-Ergebnisse der Gegenstände, die manuell segmentiert wurden, werden bei einem erneuten Start der automatischen Segmentierung nur nach einer Abfrage überschrieben.
Wählt man in dem „Segmentierungs-Übersicht“-Assistenten (siehe Abbildung 9.12) die Option `Manuell segmentieren`, gelangt man zu dem in Abbildung 9.16 dargestellten Assistenten. Hier gibt es für jedes Objekt (Stab, Prothese, Diamant, Punktzuordnung und Achsen) eine eigene Segmentierroutine, die durch Anklicken der Reiter ausgewählt werden können. In Abbildung 9.16 sind nicht alle Reiter für die einzelnen Segmentierroutinen zu sehen. Mit Hilfe der Pfeile (in Abbildung 9.16 mit (1) gekennzeichnet) kann zwischen den einzelnen Segmentierroutinen gewechselt werden.
Die Segmentierroutinen für die einzelnen Objekte werden nachfolgend detailliert beschrieben.

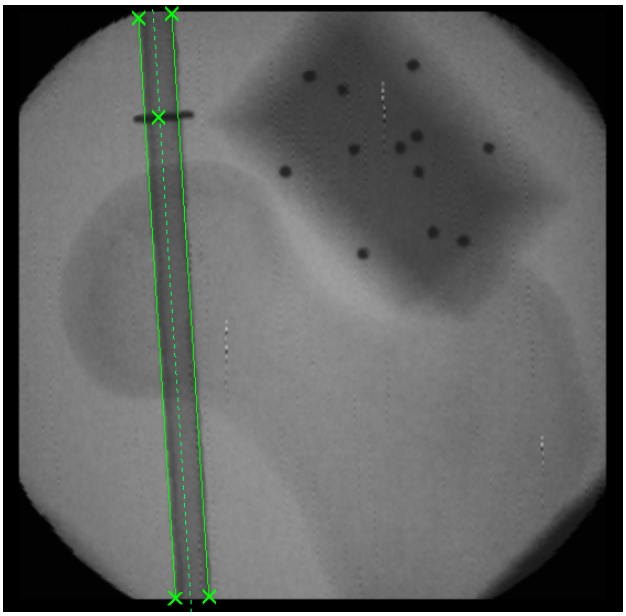
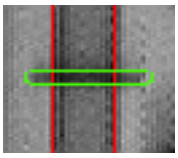


Abbildung 9.17: Korrekte Segmentierung des Stabes

Stab



Der „Stab-Segmentierungs“-Assistent ist in Abbildung 9.16 zu sehen. Der Stab (in der nebenstehenden Abbildung rot markiert) und dessen Markierung (grün umrandet) müssen in jedem Bild segmentiert werden. Dafür stehen folgende Funktionen zur Verfügung:

Linke bzw. rechte Stabkante

Zum Markieren einer Stabkante muss zunächst der Button `Punkt oben` der entsprechenden Kante angeklickt werden. Anschließend kann der obere Stab-Punkt an die linke bzw. rechte obere Kante des Stabes mit der linken Maustaste gesetzt werden. Um den unteren Stab-Punkt einzeichnen zu können, muss der Button `Punkt unten` angeklickt werden. Im Anschluss daran kann der Stab-Punkt an die untere Kante des Stabes mit der linken Maustaste gesetzt werden und die segmentierte Kante wird in der Röntgen-Aufnahme eingezeichnet.

Markierung

Betätigt man den `Markierungs`-Button kann in der Fluoroskopieaufnahme der Punkt auf die Markierung des Stabes gesetzt werden.

Stab löschen

Mit dem `Stab löschen`-Button wird die gesamte Segmentierung für den Stab (linke und rechte Stabmarke sowie die Markierung) gelöscht.

Eine korrekt eingezeichnete Markierung des Stabes ist in Abbildung 9.17 zu sehen. Die grünen Kreuze markieren die angeklickten Punkte, die grünen Linien sollen sich möglichst mit den Stabkanten decken. Die gestrichelte grüne Linie zeigt die berechnete Mittellinie des Stabes.

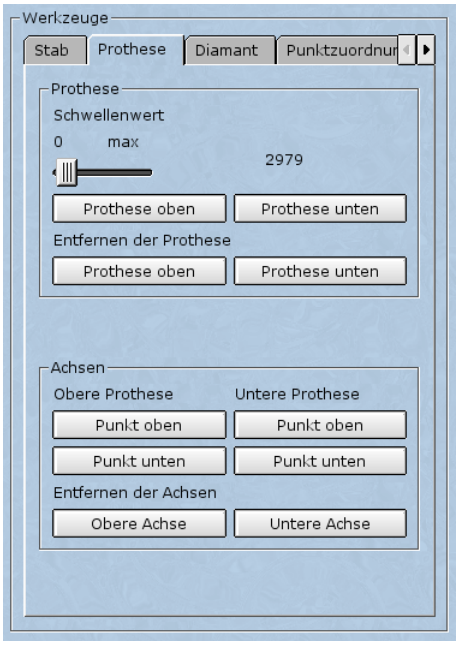


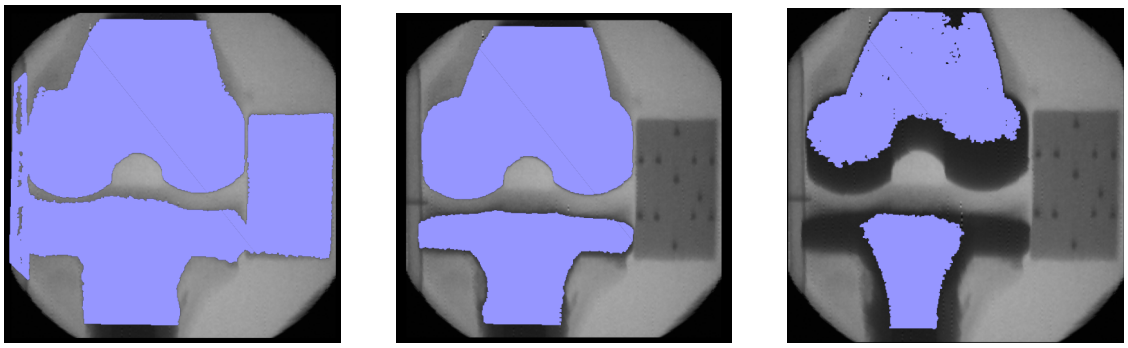
Abbildung 9.18: Manuelle Segmentierung der Prothese

Prothese
In Abbildung 9.18 ist der Assistent zur manuellen Segmentierung der Prothese zu erkennen. Zur Segmentierung der Prothese stehen die nachfolgend aufgeführten Funktionen zur Verfügung.

Hinweis:
Da die Prothese nur auf den Knie-Bildern zu sehen ist, kann auch nur hier die Prothese segmentiert werden. Daher ist für den Fall, dass in der Combobox nicht `Knie` oder `Alle` ausgewählt wurde, die Segmentierung der Prothese ausgeschaltet.

Wahl des Schwellenwertes
Bei der Segmentierung der Prothese kann ein Schwellenwert gesetzt werden. Mit dem Schwellenwert wird der Grauwert definiert, bis zu dem alle dunkleren Werte als zur Prothese gehörend definiert werden. Somit werden bei einem niedrigen Schwellenwert nur sehr dunkle Töne als zu der Prothese gehörend erkannt, während bei einem hohen Schwellenwert auch helle Grautöne mit segmentiert werden. Es ist meistens erforderlich, dass der Benutzer verschiedene Schwellenwerte ausprobiert um ein optimales Ergebnis zu erzielen. Abbildung 9.19 zeigt Beispiele für eine gute bzw. schlechte Segmentierung.

Segmentieren der Prothese
Betätigt man nach der Wahl des Schwellenwertes den Knopf `Prothese oben` bzw. `Prothese unten` und klickt anschließend mit der linken Maustaste auf die obere bzw. untere Prothese so wird diese segmentiert.



(a) Schwellenwert zu groß (b) passender Schwellenwert (c) Schwellenwert zu klein

Abbildung 9.19: Beispiele für die Wahl des Schwellenwertes bei der Prothesensegmentierung

Entfernen

Durch Anklicken des `Prothese oben`- bzw. `Prothese unten`-Buttons zum Entfernen der Prothese wird die Segmentierung der oberen bzw. unteren Prothese gelöscht.

Einzeichnen der Prothesenachsen

Für das Einzeichnen der Prothesenachsen stehen für die obere und die untere Prothese getrennte Buttons zur Verfügung. Nach Betätigung des `Punkt oben`-Buttons kann der Achsen-Punkt an dem oberen Rand der entsprechenden Prothese mit der linken Maustaste gesetzt werden. Wird anschließend der `Punkt unten`-Button angeklickt und der entsprechende Achsenpunkt in der Fluoroskopieaufnahme mit der linken Maustaste gesetzt, so wird die Achse in die 2D-Ansicht eingezeichnet. Abbildung 9.20 zeigt ein Beispiel mit eingezeichneten Prothesenachsen.

Entfernen der Achsen

Durch Klicken des Buttons `Obere Achse` bzw. `Untere Achse` kann die Achse der oberen bzw. unteren Prothese entfernt werden.

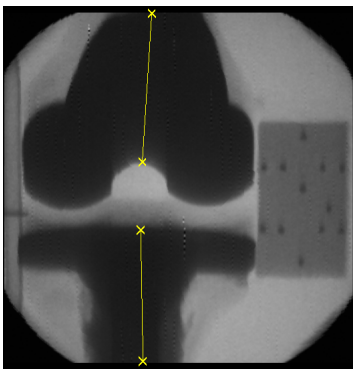


Abbildung 9.20: eingezeichnete Prothesenachsen

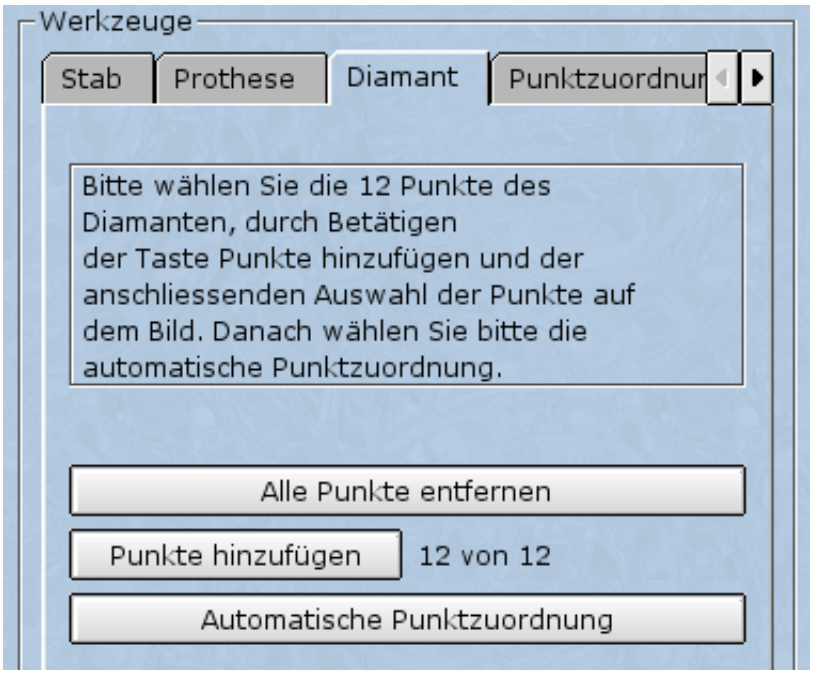


Abbildung 9.21: Assistent: Manuelle Segmentierung des Diamanten

Diamant
In Abbildung 9.21 ist der Assistent für die manuelle Segmentierung des Diamanten zu sehen. Es stehen folgende Optionen zur Verfügung:

Aktuelle Punkte entfernen
Die Punkte des Diamanten des aktuellen Bildes werden entfernt. Dies ist notwendig, wenn die automatische Segmentierung nicht erfolgreich war und beispielsweise nicht alle Punkte des Diamanten gefunden oder falsche Diamantpunkte erkannt wurden. In diesen Fällen ist eine manuelle Korrektur erforderlich. Nachdem die Punkte entfernt wurden, können sie mit Hilfe der Maus neu positioniert werden (siehe Button `Punkte hinzufügen`).

Punkte hinzufügen
Nach Betätigen des Buttons `Punkte hinzufügen` können im angezeigten Bild die Punkte des Diamanten angeklickt werden. Es müssen alle zwölf Punkte (siehe Abbildung 9.22) des Diamanten markiert werden. Für ein optimales Rekonstruktionsergebnis, ist es notwendig möglichst genau die Mittelpunkte der kleinen Kreise anzuklicken. Die Reihenfolge, in der die Punkte angeklickt werden, ist beliebig. Falls die Punkte nicht in der korrekten Reihenfolge angeklickt wurden (die entstehende Zeichnung bildet nicht den erwünschten Würfel) sollte der Button `Automatische Punktzuordnung` angeklickt werden.

Automatische Punktzuordnung
Aufgrund der besonderen Anordnung der Punkte im Diamanten können die Punkte eindeutig zugeordnet werden. Nach Betätigen des Buttons `Automatische Punktzuordnung` werden die angeklickten Punkte sortiert und es entsteht ein Diamant so wie in Abbildung 9.22 zu sehen ist. Falls die Punkte sehr dicht nebeneinander liegen und nur ungenau

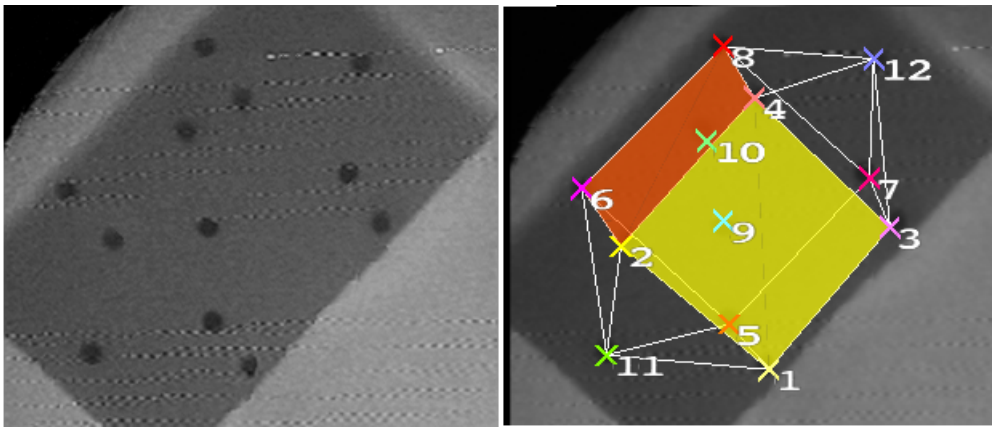


Abbildung 9.22: Manuelle Segmentierung des Diamanten

angeklickt wurden, kann unter Umständen keine automatische Punktzuordnung erfolgen. In diesem Fall muss die Zuordnung der Würfelpunkte manuell durchgeführt werden, wie im folgenden Abschnitt „Punktzuordnung“ beschrieben wird. Da die automatische Zuordnung nicht immer ein korrektes Ergebnis liefert, ist es erforderlich, dass die Ergebnisse mit Hilfe des Punktzuordnungs-Assistenten (siehe Abbildung 9.23) überprüft werden und die Zuordnung bei Bedarf korrigiert wird.

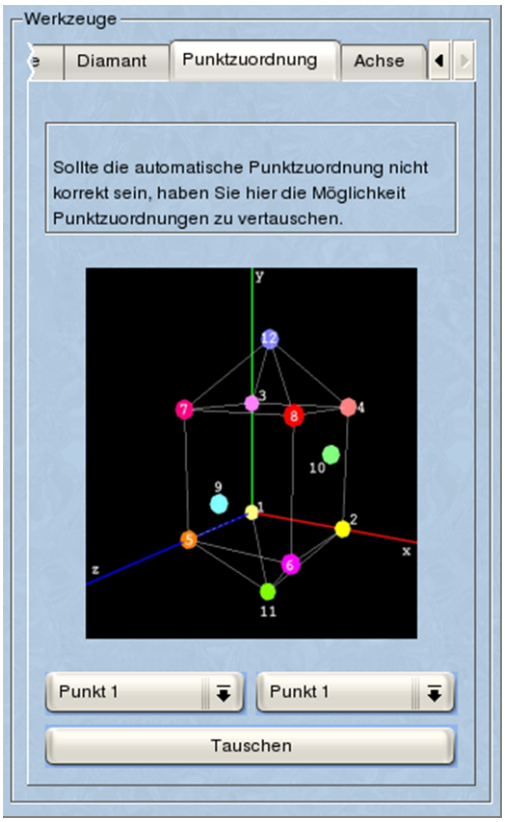


Abbildung 9.23: Manuelle Punktzuordnung

Punktzuordung
Wurden die Diamant-Punkte korrekt gesetzt, konnten aber nicht automatisch zugeordnet werden, so können die Punkte in dem in Abbildung 9.23 zu sehenden Assistenten manuell sortiert werden. In der Graphik (Abbildung 9.23) ist der Diamant mit den beschrifteten Punkten zu sehen. Die Bezeichnung der Punkte muss nun auf die zweidimensionale Aufnahme des Diamanten übertragen werden.

Comboboxen
In den beiden Auswahlfeldern (in Abbildung 9.23 mit den Aufschriften “Punkt 1“ bezeichnet) können zwei Punkte gewählt werden, die miteinander getauscht werden sollen.

Tauschen
Die in den Comboboxen ausgewählten Punkte werden getauscht.

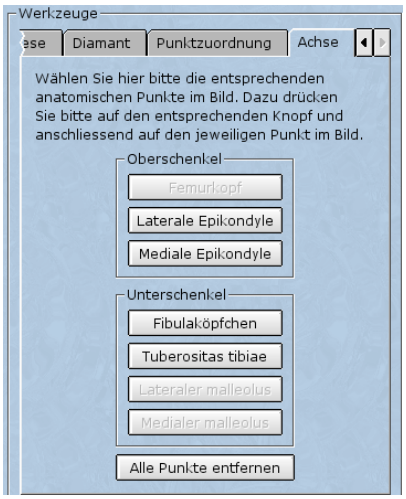


Abbildung 9.24: Manuelle Segmentierung der Achsen

Achse
Der Assistent zur Kennzeichnung der Achsen ist in Abbildung 9.24 zu sehen. Zur Definition der Achsen wird das Achsenmodell von Grood und Suntay [1] benutzt, welches die Achsen über drei Hilfsebenen rekonstruiert. Um diese drei Ebenen ermitteln zu können, müssen in den drei Sequenzen bestimmte Punkte markiert werden. In Abbildung 9.24 sind die Buttons für die Punkte aktiv, die zur Segmentierung der Achsen in der Knie-Sequenz nötig sind. Insgesamt müssen folgende Punkte in den Sequenze markiert werden:

Hüfte

- Mittelpunkt des Femurkopfes (fh)

Knie

- Laterale Epikondyle (le)
- Mediale Epikondyle (me)
- Spitze des Fibulaköpfchen (hf)
- Erhebung des Tuberositas tibiae (tt)

Knöchel

- Lateraler malleolus (lm)
- Medialer malleolus (mm)

In der Abbildung 9.25 sind die Punkte mit den entsprechenden Abkürzungen eingezeichnet.

[1] Grood, E.S., Suntay, W.J.: „A Joint Coordinate System for the Clinical Description of Three-Dimensional Motions: Application to the Knee“, Journal of Biomedical Engineering, Band 105, S. 136-144, 1983

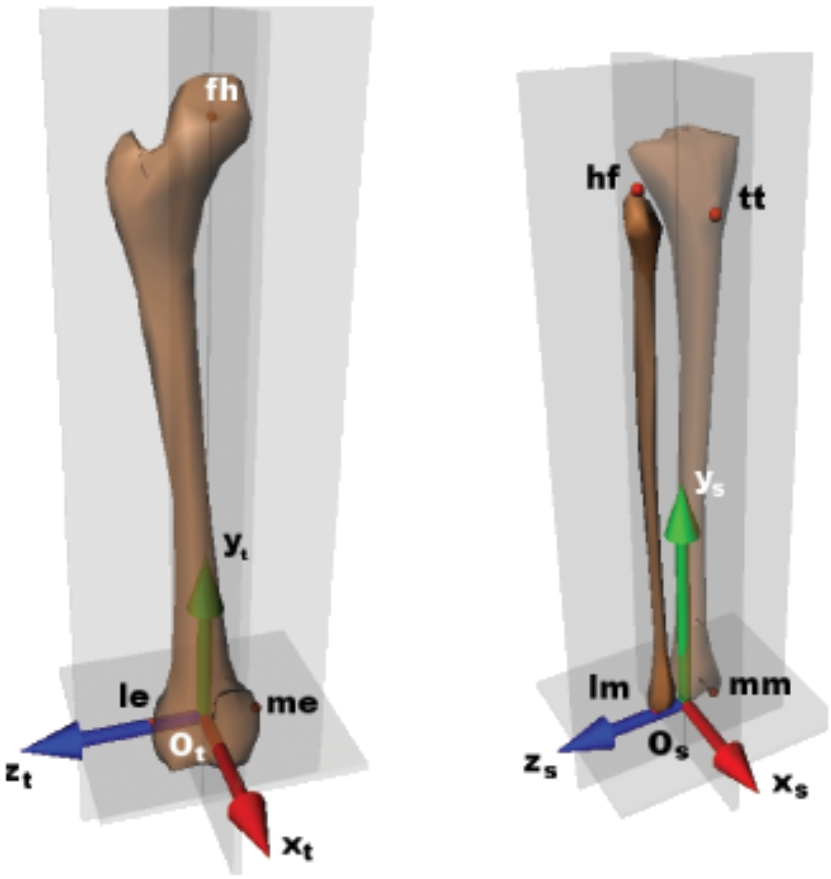


Abbildung 9.25: Relevante anatomische Punkte des Ober- und Unterschenkels

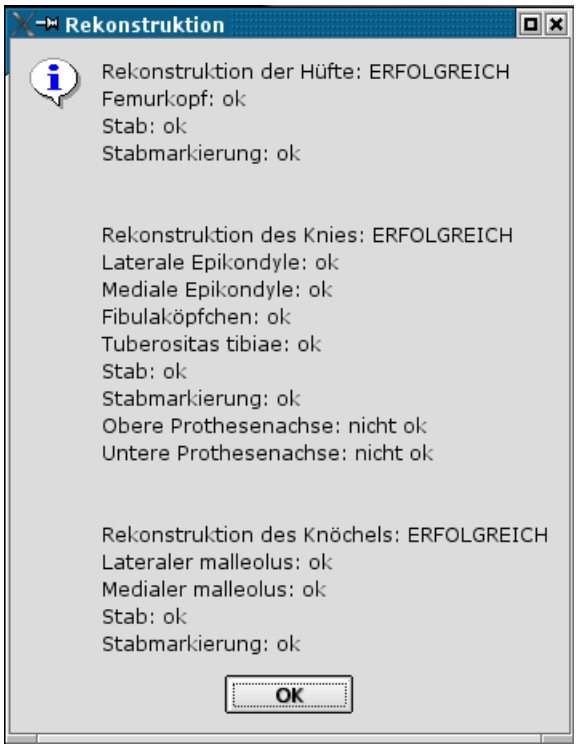


Abbildung 9.26: Erfolgs-Anzeige bei der 3D-Rekonstruktion

9.3.5 3D-Rekonstruktion

Nach dem Abschluss der Segmentierung kann die 3D-Rekonstruktion gestartet werden. Hierbei ist wichtig, dass mindestens drei Bilder von jeder Teilsequenz (Hüfte, Knie, Knöchel) vollständig segmentiert wurden. In Abbildung 9.26 wird angezeigt, welche anatomischen Bereiche erfolgreich 3D-rekonstruiert werden konnten. Hier ist in der Knie-Sequenz zu erkennen, dass beide Prothesen-Achsen (oben und unten) nicht rekonstruiert werden konnten. Dies ist gekenzeichnet durch „nicht ok“. Dieser Hinweis ist im Normalfall darauf zurückzuführen, dass die entsprechenden Punkte bzw. Achsen noch nicht segmentiert wurden.

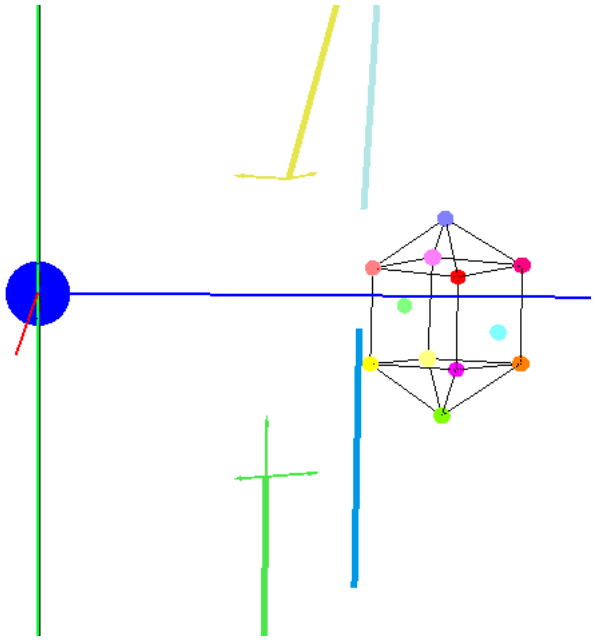


Abbildung 9.27: Gesamte 3D-Ansicht des Knies

9.3.6 3D-Ansicht

Nach der erfolgreichen 3D-Rekonstruktion können die Achsen als 3D-Ansicht betrachtet werden. Es stehen drei unterschiedliche Ansichten zur Verfügung:

1. Gesamte 3D-Ansicht
2. 3D-Ansicht des Oberschenkels
3. 3D-Ansicht des Unterschenkels

Gesamte 3D-Ansicht
In Abbildung 9.27 ist eine gesamte 3D-Ansicht des Knies zu sehen. In der Ansicht ist Folgendes dargestellt:

Stab
Auf der langen Achse sind die rekonstruierten Teile des Stabes in weiß dargestellt, die Markierungen des Stabes in blau.

Diamanten
Zur Orientierung werden die rekonstruierten und aneinander ausgerichteten Diamanten der drei Teilbereiche eingezeichnet.

Prothesen-Achsen
Die obere rekonstruierte Prothesen-Achse ist türkis dargestellt, die untere blau.

Oberschenkel-/Unterschenkel-Achse
Die Oberschenkel-Achse ist inkl. ihrem Koordinatensystem grün, die Unterschenkel-Achse inkl. Koordinatensystem gelb eingezeichnet

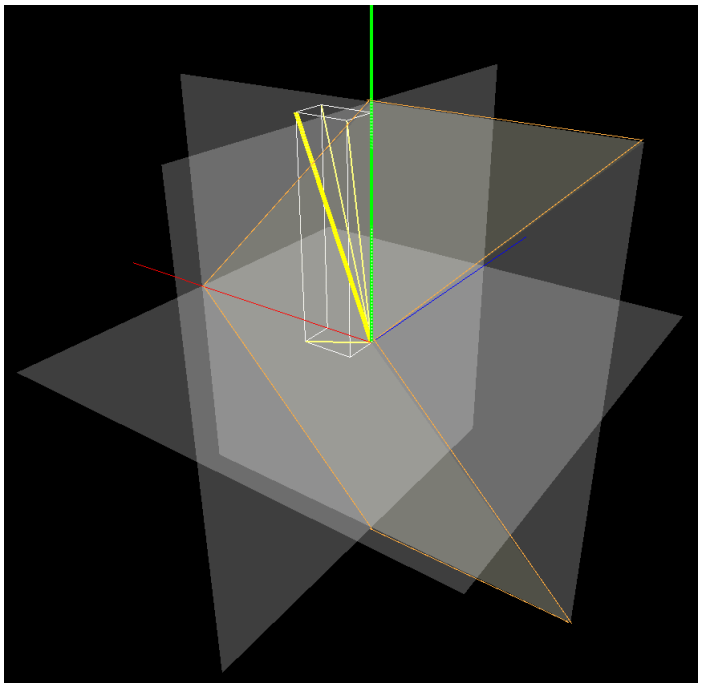


Abbildung 9.28: 3D-Ansicht des Oberschenkels

3D-Ansicht des Oberschenkels
Abbildung 9.28 zeigt eine 3D-Ansicht des Oberschenkels mit den folgenden Objekten bzw. Achsen:

Oberschenkel-Achse
Die Oberschenkel-Achse wird durch die grüne Linie repräsentiert. Darüberhinaus sind die Ebenen, die vom Koordinatensystem des Oberschenkels aufgespannt werden, eingezeichnet. Als Orientierungshilfe ist in einer Ebene ein Pfeil (in Abbildung 9.28 orange umrandet) eingezeichnet, der zur Kniescheibe hin zeigt.

Obere Prothesen-Achse
Die gelbe, fett gezeichnete Linie repräsentiert die obere Prothesen-Achse. Zusätzlich stellen die dünneren, gelben Linien die Projektion der Prothesen-Achse auf die eingezeichneten Ebenen des Oberschenkels dar.

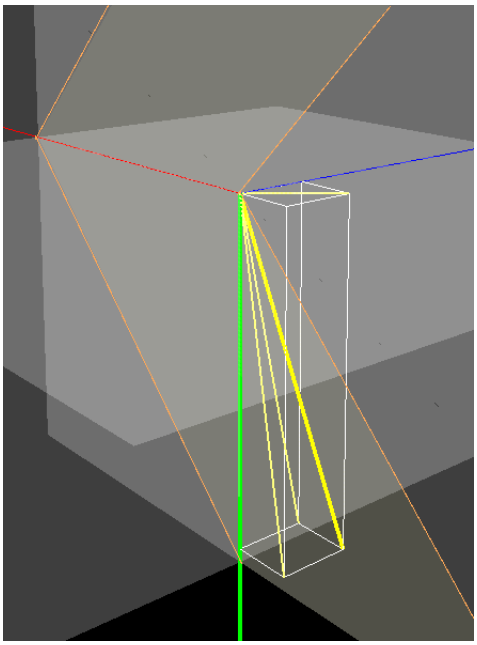


Abbildung 9.29: 3D-Ansicht des Unterschenkels

3D-Ansicht des Unterschenkels
Abbildung 9.29 zeigt eine 3D-Ansicht des Unterschenkels mit den folgenden Objekten bzw. Achsen:

Unterschenkel-Achse
Die Unterschenkel-Achse wird durch die grüne Linie repräsentiert. Darüberhinaus sind die Ebenen, die vom Koordinatensystem des Unterschenkels aufgespannt werden, eingezeichnet. Als Orientierungshilfe ist in einer Ebene ein Pfeil (in Abbildung 9.29 orange umrandet) eingezeichnet, der zur Kniescheibe hin zeigt.

Untere Prothesen-Achse
Die gelbe, fett gezeichnete Linie repräsentiert die untere Prothesen-Achse. Zusätzlich stellen die dünneren, gelben Linien die Projektion der Prothesen-Achse auf die eingezeichneten Ebenen des Unterschenkels dar.

Bedienung der 3D-Ansicht:

1. **Verschieben:**
   Mit der linken Maustaste kann das Bild verschoben werden. Dazu hält man die linke Maustaste gedrückt und bewegt die Maus in die gewünschte Richtung.

2. **Rotieren:**
   Mit der mittleren Maustaste kann die 3D-Ansicht um die Achse gedreht werden. Sollte keine mittlere Maustaste vorhanden sein, so kann auch durch gleichzeitiges Drücken der rechten und linken Maustaste derselbe Effekt erzielt werden.

3. **Zoomen:**
   Hält man die rechte Maustaste gedrückt und bewegt die Maus nach vorne bzw. hinten wird in die 3D-Ansicht rein- bzw. rausgezoomt.

4. **Wechsel zwischen den drei Ansichten:**
   Mit der Leertaste kann zwischen den drei oben beschriebenen Ansichten gewechselt werden.

9.3.7 2D-Ansicht

In der 2D-Ansicht werden die Röntgenaufnahmen mit den vorhandenen Segmentierungs-Ergebnissen angezeigt. Nach einer erfolgreichen 3D-Rekonstruktion werden auch die rekonstruierten Prothesen- sowie die Ober- und Unterschenkel-Achsen angezeigt.

Bedienung:

1. **Verschieben:**
   Mit der linken Maustaste kann das Bild verschoben werden. Dazu hält man die linke Maustaste gedrückt und bewegt die Maus in die gewünschte Richtung.

2. **Zoomen:**
   Hält man die rechte Maustaste gedrückt und bewegt die Maus vor und zurück wird in die 2D Ansicht rein und raus gezoomt.

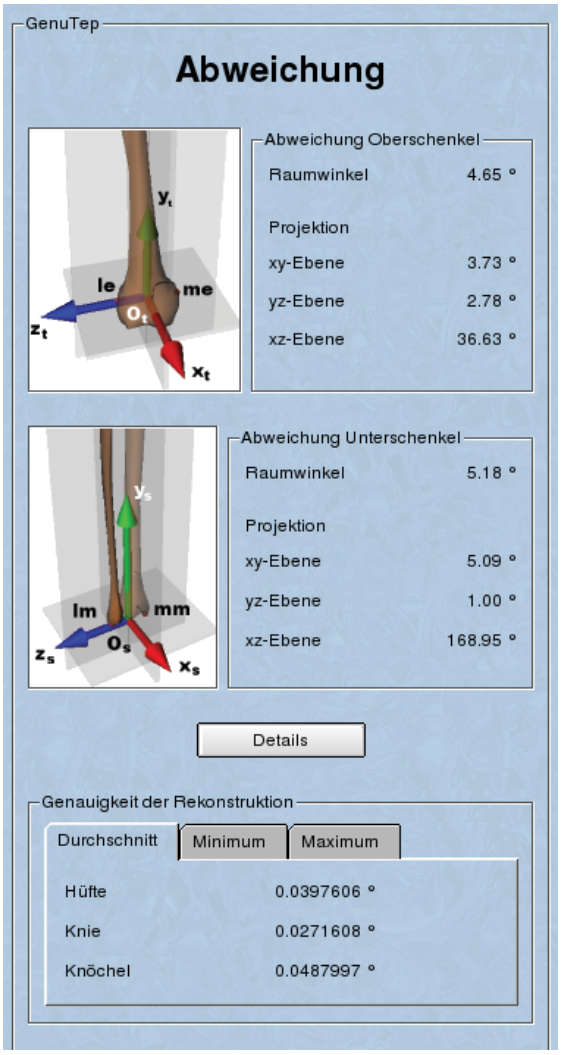


Abbildung 9.30: Abweichung

9.3.8 Abweichung

Die Abweichungen zwischen Prothese und Ober- und Unterschenkel und die Genauigkeit bei der 3D-Rekonstruktion werden in dem in Abbildung 9.30 dargestellten Fenster angezeigt.

Abweichung Oberschenkel

- **Raumwinkel:** Gibt den Winkel zwischen der oberen Prothesen-Achse und der Oberschenkel-Achse im Raum an´.
- **xy-Ebene:** Gibt den auf die xy-Ebene projizierten Winkel zwischen der oberen Prothesen-Achse und der Oberschenkel-Achse an.
- **yz-Ebene:** Gibt den auf die yz-Ebene projizierten Winkel zwischen der oberen Prothesen-Achse und der Oberschenkel-Achse an.
- **xz-Ebene:** Gibt den auf die xz-Ebene projizierten Winkel zwischen der oberen Prothesen-Achse und der in der Abbildung eingezeichneten x-Achse an.

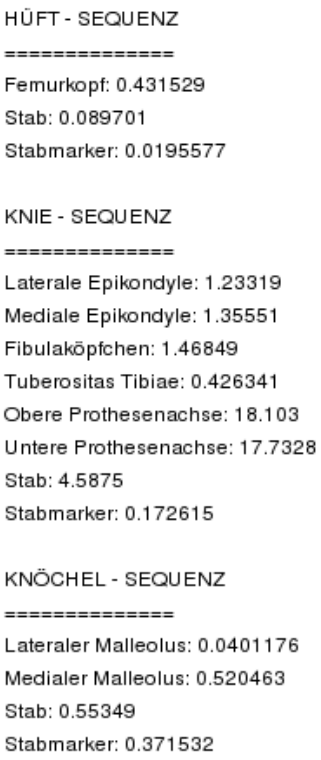


Abbildung 9.31: Genauigkeit der Rekonstruktion

Abweichung Unterschenkel

- **Raumwinkel:** Gibt den Winkel zwischen der unteren Prothesen-Achse und der Unterschenkel-Achse im Raum an.
- **xy-Ebene:** Gibt den auf die xy-Ebene projizierten Winkel zwischen der unteren Prothesen-Achse und der Unterschenkel-Achse an.
- **yz-Ebene:** Gibt den auf die yz-Ebene projizierten Winkel zwischen der unteren Prothesen-Achse und der Unterschenkel-Achse an.
- **xz-Ebene:** Gibt den auf die xz-Ebene projizierten Winkel zwischen der unteren Prothesen-Achse und der in der Abbildung eingezeichneten x-Achse an.

Genauigkeit Rekonstruktion

Gibt die Genauigkeit mit der die Rekonstruktion durchgeführt werden konnte an (in Prozent).

Details

Zeigt detailliert die Genauigkeiten der Rekonstruktion der einzelnen Punkte bzw. des Stabes an (siehe Abbildung 9.31).

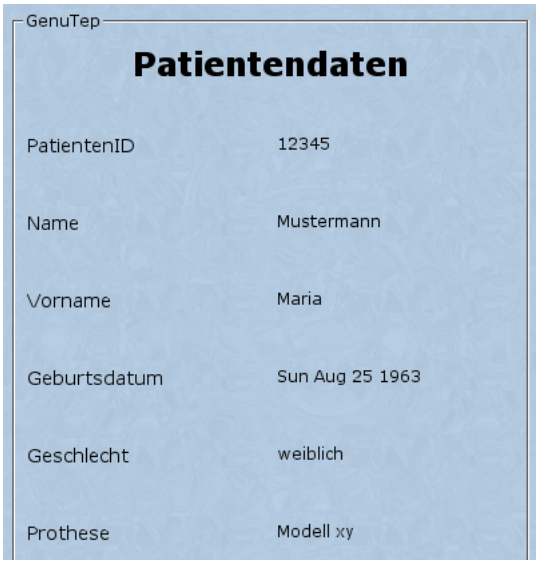


Abbildung 9.32: Patientendaten

9.3.9 Patientendaten

Durch Anklicken des entsprechenden Buttons in der Toolbar des Hauptfensters gelangt man zum „Patientendaten“-Dialog (siehe Abbildung 9.32), in dem die Patientendaten angezeigt werden, die beim Anlegen des Projektes angegeben bzw. ausgelesen wurden (siehe Kapitel 9.3.1.3). Wahlweise kann dieser Dialog auch über den Menü-Eintrag `Ansicht/ Patientendaten` aufgerufen werden. Die Patientendaten können in diesem Dialog jedoch nicht mehr verändert werden.

9.4 Anhang

9.4.1 Kalibrieren der Kamera

9.4.1.1 Installation

Das Kalibrierungsmodul wird als Tar-Archiv geliefert. Die Installation erfolgt Linux-typisch (s.u.). Zum Kompilieren muss die Mathematik-Bibliothek Newmat (ab Version 9) verfügbar sein. Die Bibliothek wird bei der Übersetzung statisch eingebunden. Dazu muss die Umgebungsvariable NEWMAT auf den Pfad der Bibliothek gesetzt werden. Unter bash als shell geschieht dies folgendermaßen:

```
NEWMAT=<Pfad zur NewMat Bibliothek>
export NEWMAT
```

Die Installation verläuft in folgenden Schritten:

- Entpacken des Archivs in ein Verzeichnis
- Setzen der Umgebungsvariablen für NEWMAT
- ./configure
- make install

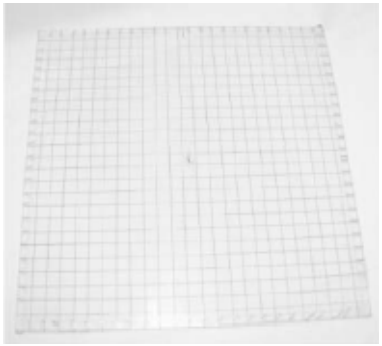


Abbildung 9.33: Kalibrierungs-Gitter

9.4.1.2 Kalibrierungsobjekt

Als Kalibrierungsobjekt wird ein planares röntgendichtes Gitter (siehe Abbildung 9.33) mit bekannten Maßen verwendet. Die Abstände im Gitter können im Optionsdialog eingestellt werden. Zu Orientierungszwecken ist auf dem Bild ein Marker zu plazieren, damit die korrespondierenden Punkte zugeordnet werden können. Es müssen mindestens 3 Aufnahmen des Gitters unter verschiedenen Winkeln (60-90°) angefertigt werden. Für jeden Bildverstärker muss eine Aufnahmenserie durchgeführt werden.

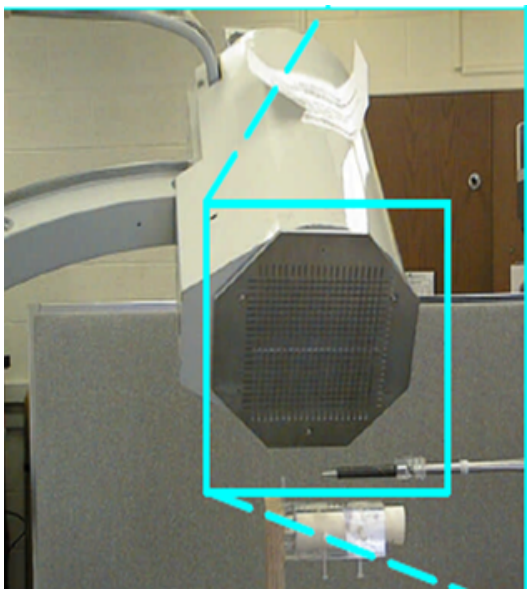


Abbildung 9.34: Anordnung des Kalibrierungsgitters

9.4.1.3 Anordnung

Das Kalibrierungsgitter wird direkt vor dem Detektor des C-Bogens (wie in Abbildung 9.34 zu sehen) befestigt. Mit dieser Anordnung wird eine Aufnahme des Gitters gemacht. Diese Aufnahme liefert wichtige Informationen für das Entzerren der Aufnahmen. Anschließend wird das Gitter auf dem Tisch platziert und es müssen aus mindestens drei unterschiedlichen Winkeln Aufnahmen von dem Gitter getätigt werden. Mit Hilfe dieser Aufnahmen können die intrinischen Parameter der Kamera ermittelt werden, die zur Kalibrierung des C-Bogens benötigt werden.

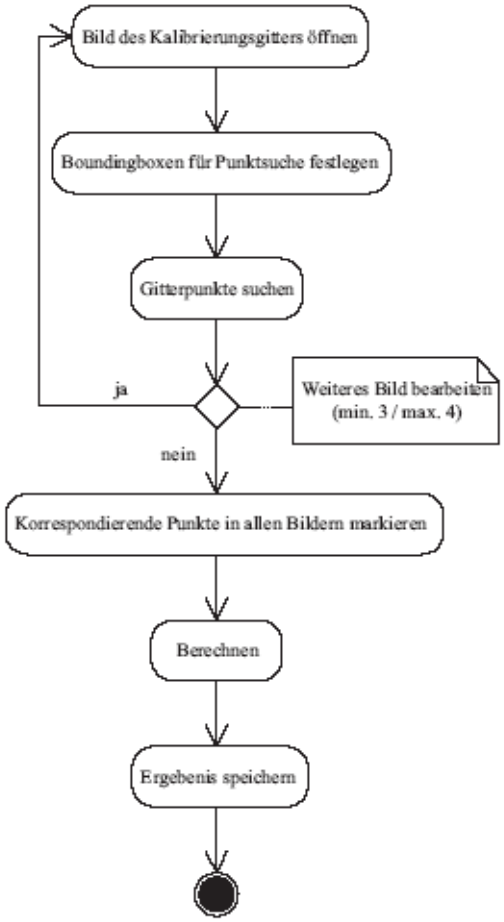


Abbildung 9.35: Ablauf der Kalibrierung

9.4.1.4 Bedienung mit Hilfe des Assistenten

Bei der Bedienung wird der Benutzer von einem Assistenten unterstützt. Dieser führt den Benutzer schrittweise durch das Programm. Die einzelnen Schritte sind in Diagramm 9.35 dargestellt. Abbildung 9.36 zeigt das Programm direkt nach dem Start.

Assistent
: Wird auf der rechten Seite angezeigt. Gibt während des gesamten Programmablaufes Hilfeleistung und Tipps.

Menü/Ansicht
: Die Anzeige der Bilder kann über das Menü `Ansicht` vergrößert bzw. verkleinert werden. Alternativ kann dies mit den Tasten „+“ und „-“ geschehen.

Assistent deaktivieren
: Der Assistent wird abgeschaltet und das Programm geht in den Experten-Modus. Der Assistent wird dann erst wieder gestartet, wenn ein neues Projekt gestartet wird.

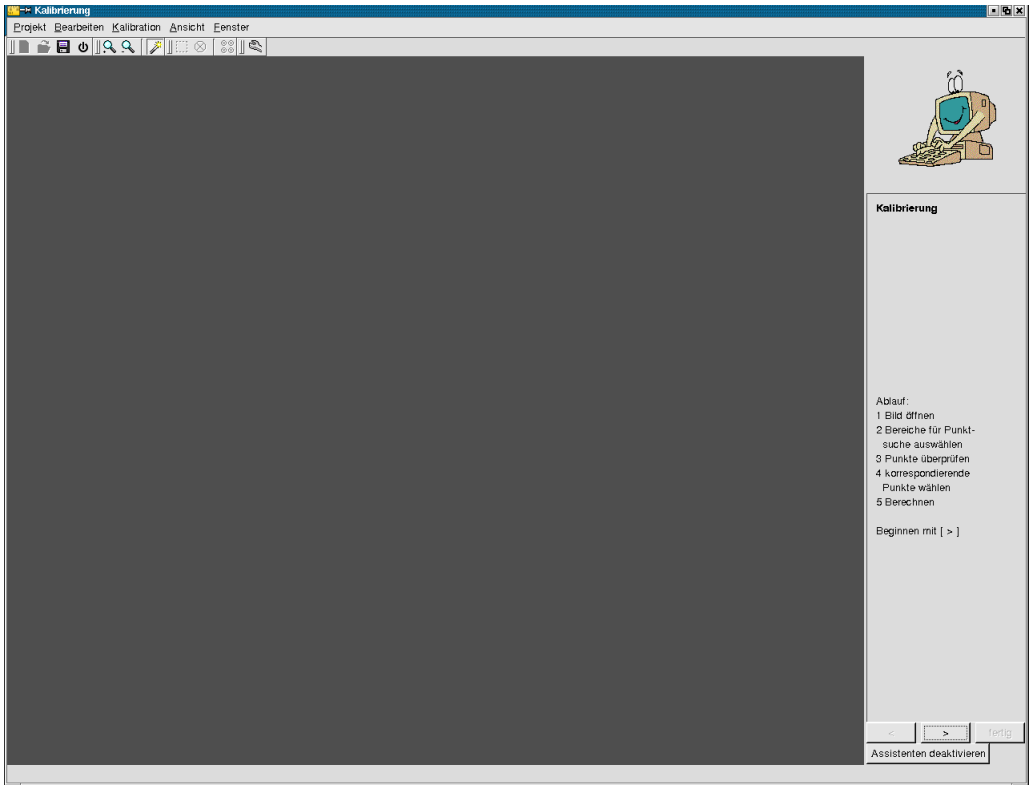


Abbildung 9.36: Kalibrierung Hauptfenster

1. **Bild öffnen**
   Zu Beginn muss ein Bild des Kalibrierungsgitters geöffnet werden. Es erscheint automatisch der „Datei öffnen“-Dialog. Hier muss ein Bild zum Öffnen ausgewählt werden.

   Open
   Bestätigung der Auswahl des Bildes. Bild wird geöffnet. Der Dialog schließt sich.

   Cancel
   Der Dialog wird geschlossen. Die Aktion wird abgebrochen.

2. **Boundingboxen setzen**
   Boundingboxen bieten die Möglichkeit, Bereiche, die bei der automatischen Gitterpunktsuche zu Problemen führen können (z.B. Marker), auszuschließen. In den Ecken wird automatisch eine Boundingbox generiert. Der Rand des Bildes wird im Bereich von 10 Pixeln nicht in die Suche mit einbezogen und muss somit nicht gekennzeichnet werden (Abbildung 9.37).

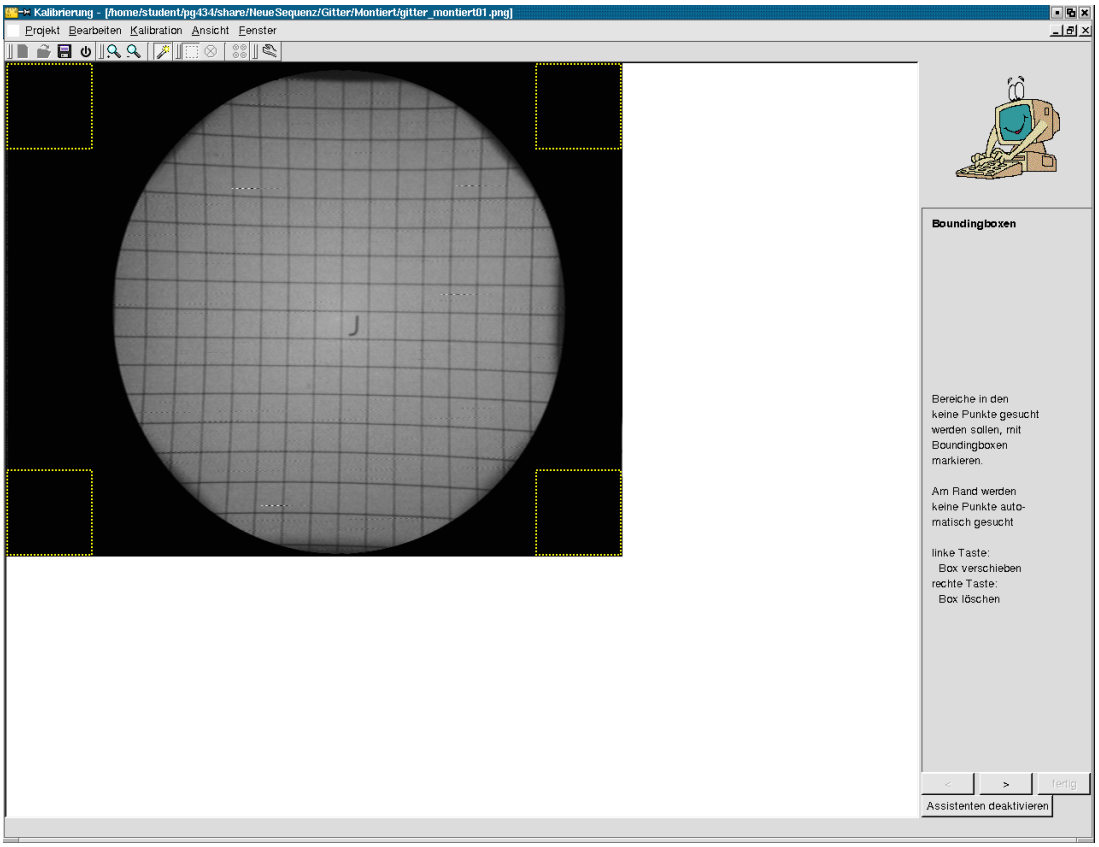


Abbildung 9.37: Boundingboxen setzen

Bedienung:

Neue Box erstellen

Mit der linken Maustaste auf die Stelle im Bild klicken, wo die neue Boundingbox beginnen soll. Durch Ziehen der Maus mit gedrückter Taste die Größe festlegen.

Eine Box verschieben

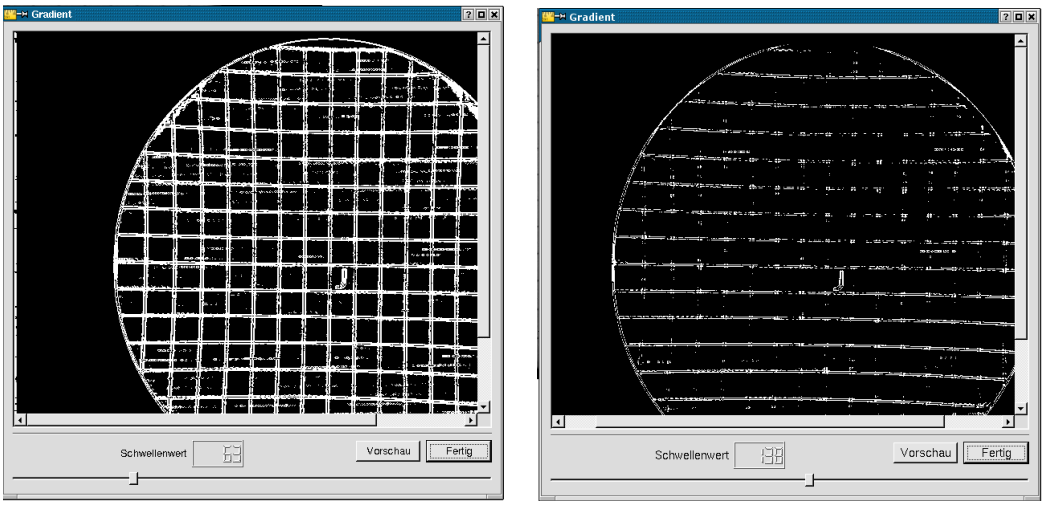
Mit der linken Maustaste auf den Rand der Boundingbox klicken. Boundingbox an die neue Position ziehen.

Eine Box vergrößern/verkleinern

Mit der linken Maustaste auf die untere rechte Ecke der Boundingbox klicken. Durch Ziehen der Maus die neue Größe festlegen.

Eine Box löschen

Den Rand der Boundingbox mit der rechten Maustaste anklicken.



(a) Guter Gradient

(b) Schlechter Gradient

Abbildung 9.38: Beispiel für einen guten bzw. schlechten Gradienten

3. **Gradientenbild bearbeiten**
   Vor der automatischen Punktsuche wird das Gradientenbild eingeblendet. Hierbei sollte der Schwellenwert so eingestellt werden, dass im Gradientenbild das Gitter gut zu erkennen ist (siehe Abbildung 9.38). In den meisten Fällen sollte die Voreinstellung von 60 ausreichende Ergebnisse liefern.

   Vorschau
   Eine Vorschau der Skelettierung kann eingesehen werden. Bei guter Einstellung des Gradientenschwellwerts sollte ein gleichmäßiges Gitter zu sehen sein (siehe Abbildung 9.39).

   Schwellenwert-Regler
   In den meisten Fällen sollte die Voreinstellung von 60 ausreichende Ergebnisse liefern.

   Fertig
   Die Einstellungen werden übernommen.

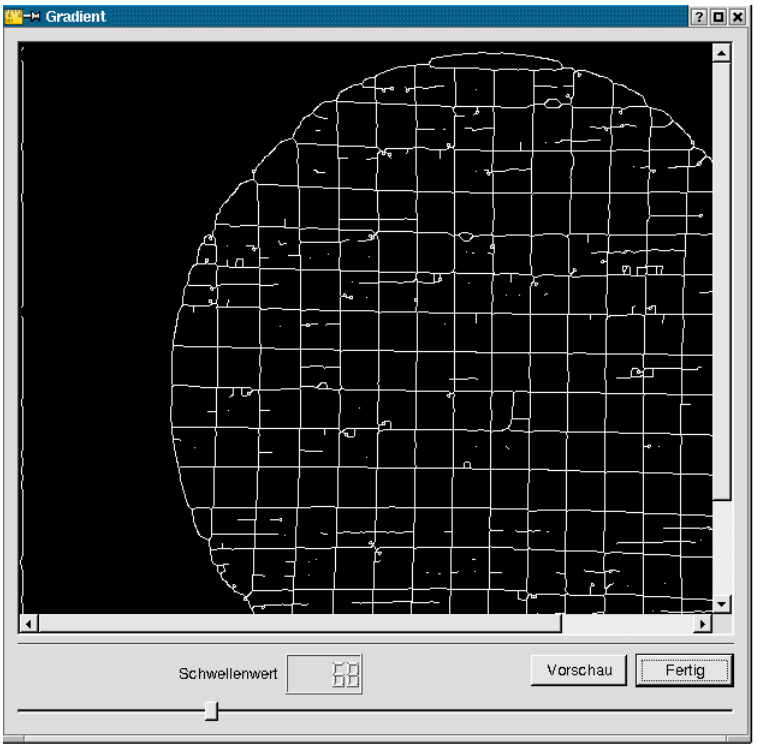


Abbildung 9.39: Vorschau nach Anwendung der Skelettierung

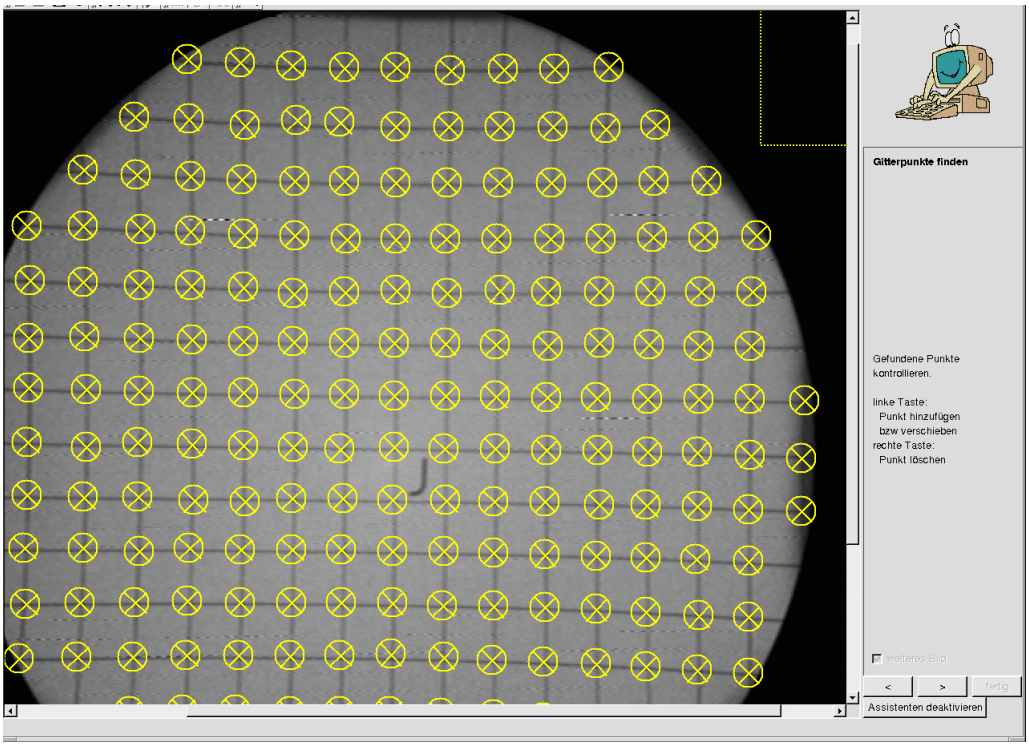


Abbildung 9.40: Ansicht nach Abschluss der Gitterpunktsuche

4. **Gitterpunkte finden bzw. korrigieren**
   Nach dem Festlegen der Boundingboxen und ggf. der Überprüfung des Gradientenbildes wird der automatische Suchalgorithmus gestartet. Nach Abschluss der Gitterpunktsuche wird das Bild mit den gefundenen Punkten angezeigt (Abbildung 9.40). In dieser Ansicht müssen die gefundenen Punkte überprüft und ggf. korrigiert werden. Fehlerquellen sind hier eventuell schlecht positionierte Marker auf dem Bild.

   Bedienung:

   Punkt hinzufügen
   Mit der linken Maustaste klicken.

   Punkt verschieben
   Mit der linken Maustaste auf Punkt klicken und ziehen.

   Punkt löschen
   Mit der rechten Maustaste auf den Punkt klicken.

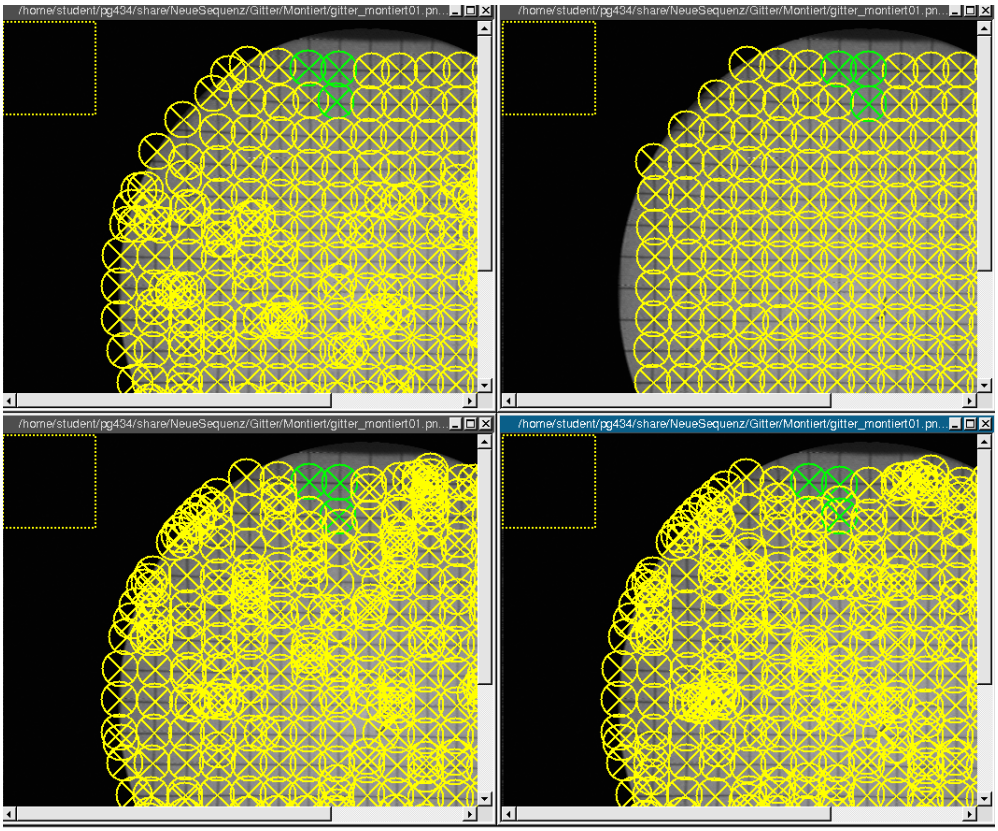
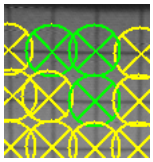


Abbildung 9.41: Ansicht nach Abschluss der Markierung der korrespondierenden Punkte

5. **Korrespondierende Punkte setzen**

   Um eine Beziehung zwischen den einzelnen Bildern herstellen zu können, müssen in allen Bildern jeweils drei korrespondierende Punkte gesetzt werden. Dazu werden in jedem Bild ein Punkt, dessen linker Nachbar sowie dessen unterer Nachbar markiert.

   Hinweis:
   Alle Bilder werden zur besseren Übersicht nebeneinander angezeigt. Es empfiehlt sich einen Punkt in der Nähe eines Markers zu markieren, da so gewährleistet ist, dass die Positionen der Punkte sich in den einzelnen Bildern entsprechen (siehe Abbildung 9.41).

6. **Berechnen**
   Sind die korrespondierenden Punkte markiert kann die Berechnung mit `Fertig` gestartet werden. Am Ende der Berechnung kann das Ergebnis in einer Datei gespeichert werden und der Assistent wird beendet.

9.4.1.5 Experten-Modus

Im Experten-Modus müssen die einzelnen Schritte der Kalibrierung selbstständig durchgeführt werden. Dieses sind im Einzelnen:

1. **Bilder auswählen**
   Zunächst müssen die C-Bogen-Aufnahmen des Kalibrierungsgitters ausgewählt werden. Dies geschieht über das Menü `Kalibrierung`/`Bild öffnen`. Es können Bilder der Formate JPEG und PNG geladen werden. Es sind mindestens drei Bilder zur Berechnung erforderlich, maximal können vier Bilder bearbeitet werden.

2. **Gitterpunkte markieren**
   Zur Markierung der Gitterpunkte empfiehlt es sich zunächst Boundingboxen festzulegen. Mit Hilfe von Boundingboxen können Bereiche markiert werden, die in der automatischen Punktsuche übersprungen werden sollen. Grundsätzlich kann in zwei Modi gearbeitet werden: Boundingboxen bearbeiten (zu erreichen über das Menü `Bearbeiten`/`Boundingboxen`) und Gitterpunkte bearbeiten (zu erreichen über das Menü `Bearbeiten`/`Gitterpunkte`).

   (a) Boundingboxen festlegen
   (b) automatische Gitterpunktsuche starten
   (c) Gitterpunkte überprüfen

   Die automatische Gitterpunktsuche wird über das Menü `Kalibrierung`/`Gitterpunkte suchen` gestartet. Über den Eintrag `Kalibrierung`/`Gitterpunkte löschen` können alle Punkte gelöscht werden.

   Die Steuerung von Boundingboxen und Gitterpunkten:

   Boundingbox - Hinzufügen:
   Mit linker Maustaste auf einen freien Bereich klicken und ziehen.

   Boundingbox - Verschieben:
   Mit linker Maustaste auf den Rand klicken und ziehen.

   Boundingbox - Größe ändern:
   Mit der linken Maustaste auf die untere rechte Ecke klicken und ziehen.

   Boundingbox - Löschen:
   Mit der rechten Maustaste auf den Rand klicken.

   Gitterpunkte - Hinzufügen:
   Mit der linken Maustaste auf einen freien Bereich klicken.

   Gitterpunkte - Verschieben:
   Mit der linken Maustaste auf einen Punkt klicken und ziehen.

   Gitterpunkte - Löschen:
   Mit der rechten Maustaste auf einen Punkt klicken.

3. **korrespondierende Punkte auswählen**
   Um die Bilder untereinander in Verbindung zu setzen, müssen nun in jedem Bild drei Punkte markiert werden (ein Punkt, dessen linker Nachbar und dessen unterer Nachbar). Die Position der markierten Punkte muss in allen Bildern gleich sein.

(a) Über das Menü `Bearbeiten`/`Korrespondierende Punkte` den Modus "korrespondierende Punkte" wählen.

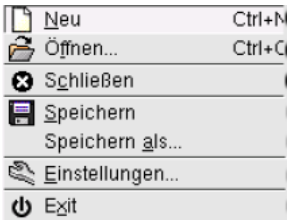
(b) Die Punkte innerhalb der Bilder anklicken. Es ist darauf zu achten, dass pro Bild immer nur drei Punkte in der oben genannten Form markiert sind (siehe Abbildung 9.41).

4. **Kalibrierung starten**
   Nachdem alle Bilder bearbeitet wurden (mindestens drei) kann die Berechnung mit `Kalibrierung`/`Kalibrieren` gestartet werden. Ist die Berechnung erfolgreich, kann die Matrix mit den intrinsischen Parametern gespeichert werden.

9.4.1.6 Menüstruktur

An dieser Stelle wird ein Überblick über die einzelnen Menü-Einträge gegeben. Im Assistenten-Modus sind nicht alle Einträge verfügbar. Über die Icons vor dem Namen können die Aktionen auch über die Toolbar ausgewählt werden.

Das Projekt-Menü
Die Einträge haben folgende Bedeutung:

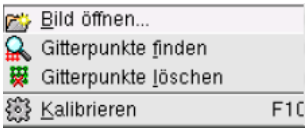


- **Neu:** Ein neues Projekt starten
- **Öffnen:** Ein zuvor gespeichertes Projekt öffnen.
- **Schließen:** Das aktuelle Projekt schließen.
- **Speichern:** Das aktuelle Projekt speichern.
- **Speichern als:** Das aktuelle Projekt unter einem neuen Namen speichern.
- **Einstellungen:** Den Optionsdialog öffnen.
- **Exit:** Das Programm beenden.

Das Bearbeiten-Menü
Die Einträge haben folgende Bedeutung:

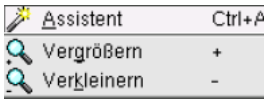
- **Gitterpunkte:** Schaltet in den Modus Gitterpunkte. In diesem Modus können Gitterpunkte bearbeitet werden.
- **Boundingboxen:** Schaltet in den Modus Boundingbox. In diesem Modus können Boundingboxen bearbeitet werden.
- **Korrespondierende Punkte:** Schaltet in den Modus Korrespondierende Punkte. In diesem Modus werden die korrespondierenden Punkte festgelegt.

Das Kalibrierung-Menü
Die Einträge haben folgende Bedeutung:



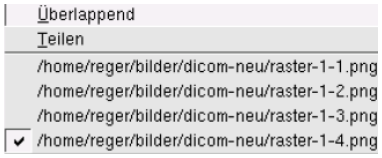
- **Bild öffnen:** Öffnet ein Bild des Kalibrierungsobjekts zur Bearbeitung.
- **Gitterpunkte finden:** Startet die automatische Gitterpunktsuche.
- **Gitterpunkte löschen:** Löscht alle Gitterpunkte.
- **Kalibrieren:** Startet die Berechnung.

Das Ansicht-Menü
Die Einträge haben folgende Bedeutung:



- **Assistent:** Blendet den Assistenten ein oder aus (nur im Assistenten-Modus).
- **Vergrößern:** Vergrößert das aktuelle Bild.
- **Verkleinern:** Verkleinert das aktuelle Bild.

Das Fenster-Menü
Die Einträge haben folgende Bedeutung:



- **Überlappend:** Die Fenster werden kaskadiert angeordnet.
- **Teilen:** Die Fenster werden nebeneinander angeordnet.

Unter den ersten zwei Menüeinträgen erscheint eine Liste der geöffneten Fenster. Das aktuelle Fenster ist mit einem Häkchen gekennzeichnet. Durch Anklicken kann hier das gewünschte Fenster ausgewählt werden.

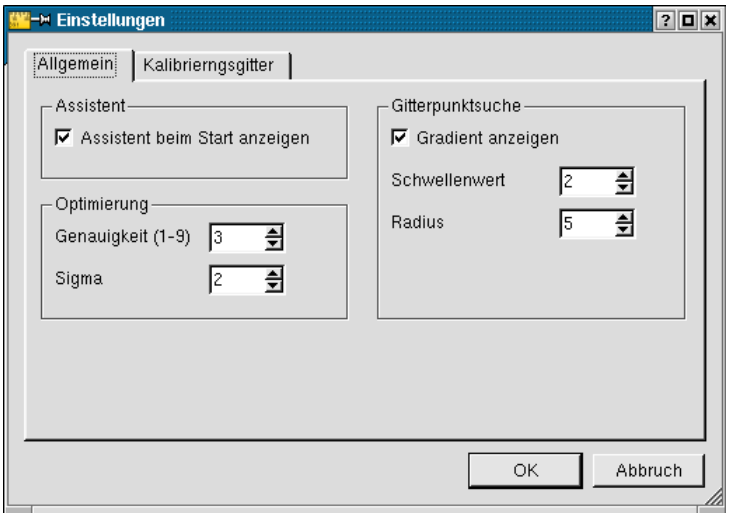


Abbildung 9.42: Optionsdialog (Tablet 1)

9.4.1.7 Einstellung

Die Abbildungen 9.42 und 9.43 zeigen den Optionsdialog. Die einzelnen Einträge haben folgende Bedeutung:

Assistent beim Start anzeigen
: Der Assistent wird beim Programmstart gestartet und führt den Anwender durch das Programm.

Genauigkeit
: Anzahl der Nachkommastellen bei der Berechnung - eine größere Zahl hat eine höhere Genauigkeit aber eine langsamere Berechnung zur Folge.

Sigma
: Standardabweichung bei der Optimierung.

Gradient anzeigen
: Vor der automatischen Gitterpunktsuche wird der Gradient angezeigt und kann bearbeitet werden.

Schwellenwert
: Voreingestellter Schwellenwert bei der Gitterpunktsuche.

Radius
: Radius, in dem Punkte bei der automatischen Gitterpunktsuche zusammengefasst werden.

horizontal in *mm*
: Horizontaler Abstand zwischen 2 Punkten im Kalibrierungsgitter in *mm*.

vertikal in *mm*
: Vertikaler Abstand zwischen 2 Punkten im Kalibrierungsgitter in *mm*.

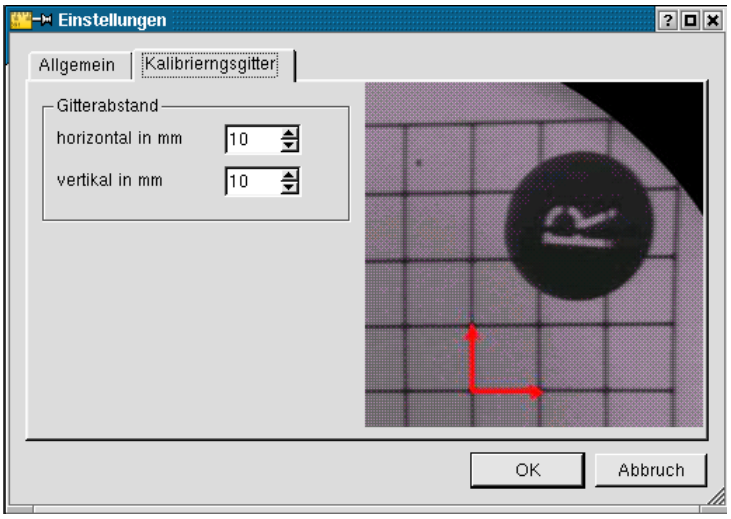


Abbildung 9.43: Optionsdialog (Tablet 2)

9.4.1.8 Speicherung der Kalibrierungs-Ergebnisse

Ist das Durchlaufen der Kalibrierung erfolgreich, wird der Benutzer aufgefordet die Ergebnisse zu speichern. Diese Ergebnisse müssen in das Verzeichnis gespeichert werden, das den Namen des neu angelegten C-Bogen trägt.

Anhang A

Klassendokumentation

A.1 Das Klassendiagramm

Das Klassendiagramm (Abbildung A.1) gibt einen Überblick über die gesamte Applikation. Die einzelenen Pakete und Klassen sind im folgenden detailierter in Kapitel A.3 beschrieben.

A.2 Paketübersicht

Wie jedes größere Softwareprojekt ist auch die GenuTEP-Applikation in verschiedene Pakete unterteilt. Abbildung A.2 gibt einen Überblick über die Paketstruktur.

A.3 Klassendokumentation

Im folgenden werden alle Klassen der GenuTEP-Anwendung kurz beschrieben. Es wird jeweils ein Diagramm des entsprechenden Paketes präsentiert. Die Dokumentation bezieht sich nur auf Pakete, Klassen, Methoden und öffentliche wie auch private Attribute. C-Structs, Enumerations, usw. werden der Übersichthalber außen vor gelassen.

A.3.1 Package 3DReconstruction

Das Paket 3DReconstruction beschäftigt sich mit der 3D-Rekonstruktion der zweidimensionalen Segmentierungsergebnisse und der Anordnung der Sequenzen der drei anatomischen Bereiche.

Klasse Reconstructor:
Die Klasse Reconstructor beinhaltet alle algorithmischen Funktionen zur dreidimensionalen Rekonstruktion. Das hier implementierte Verfahren richtet sich nach dem Algorithmus von [CH94].

Klasse Yabat:
In dieser Klasse werden die drei unabhängig voneinander rekonstruierten Szenen zueinander in Beziehung gesetzt. Dabei werden sie in einem gemeinsamen Koordinatensystem räumlich korrekt positioniert.

Klasse FatalReconstructionException:
Enthält Fehlermeldungen des Rekonstruktionsmoduls.

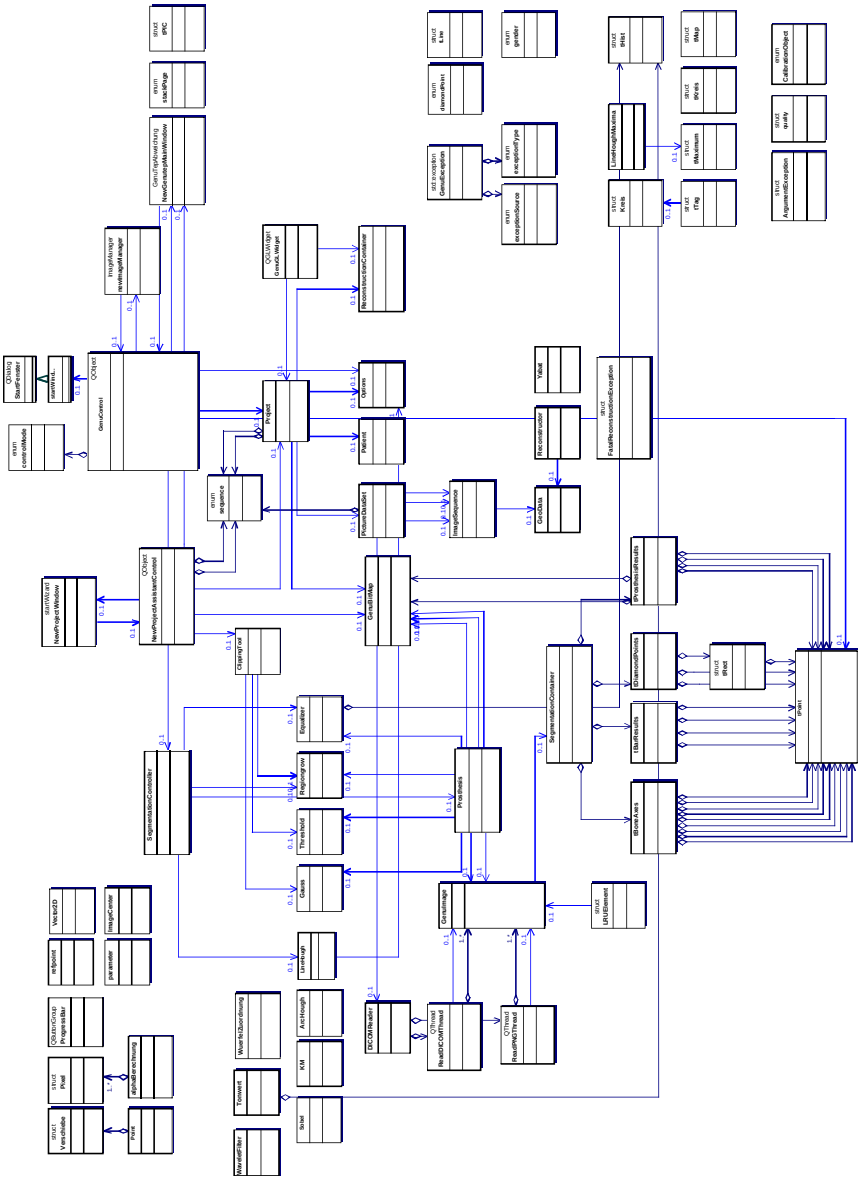


Abbildung A.1: Das Klassendiagramm der GenuTEP-Applikation

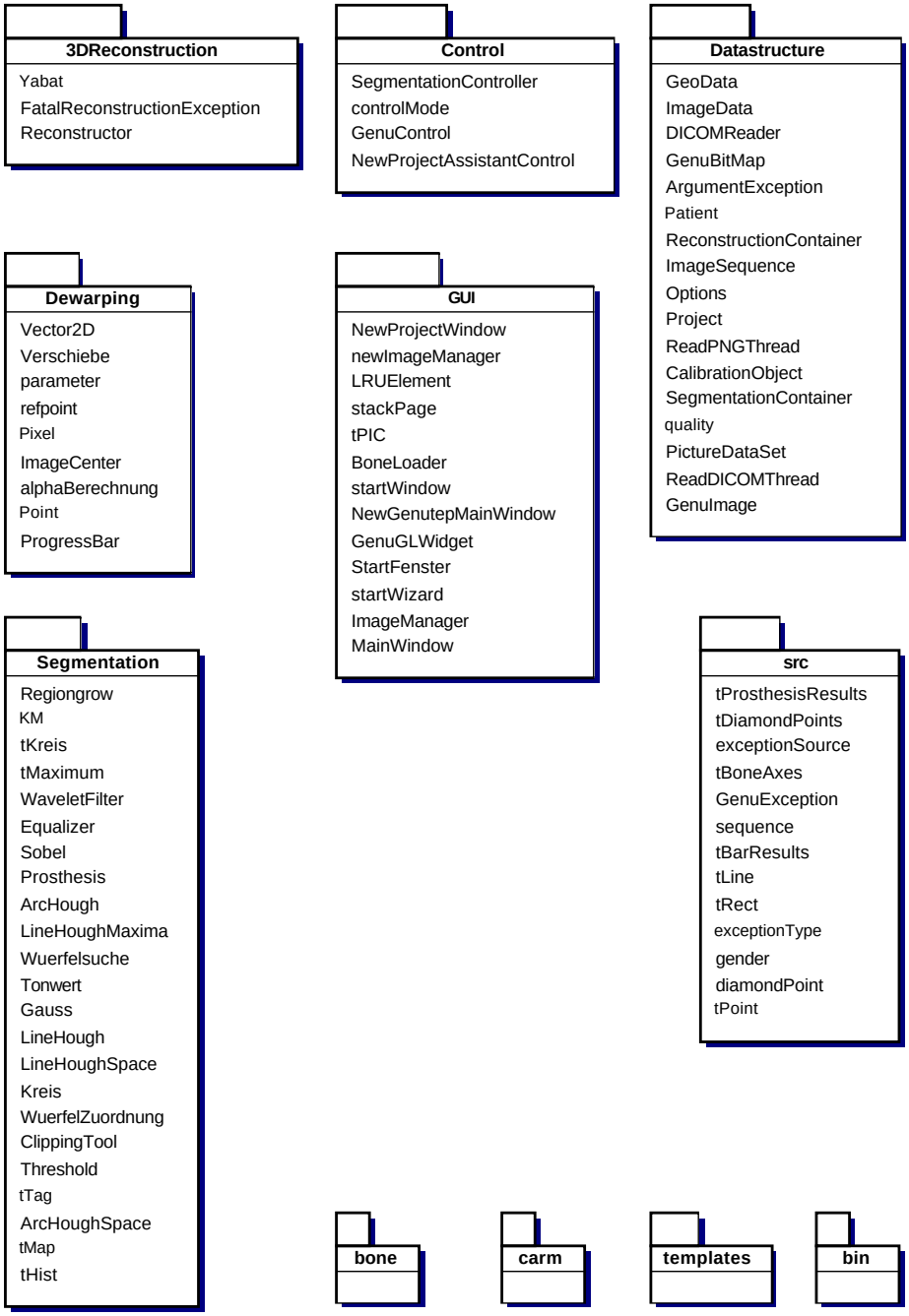


Abbildung A.2: Die Paketübersicht der GenuTEP-Applikation

Abbildung A.3: Klassendiagramm Package 3DRekonstruction

A.3.2 Package Control

Die Kontrollfunktion der Anwendung ist in den Klassen des Control-Paketes realisiert.

Klasse GenuControl:
Die Klasse GenuControl steuert den Ablauf des gesamten Programms. Alle Benutzerinteraktionen werden von dieser Klasse geprüft und die korrekten Funktionen ausgeführt.

Klasse NewProjectAssistantControl:
NewProjectAssistantControl kontrolliert den Vorgang des Projekt-Anlegens. Hierzu gehören u.a. das Einlesen und Dewarpen der Bilddaten sowie die Erstellung der Datenstruktur mit den Patienten- und globalen Projektdaten.

Klasse SegmentationController:
Der SegmentationController ist die Kontrollinstanz für die Segmentierung. Seine Methoden stehen der globalen Kontrollinstanz als Wrapper für die feingranularere Funktion der Segmentierung zur verfügung.

A.3.3 Package Datastructure

Jegliche anfallenden Daten der GenuTEP-Applikation werden in Datenstrukturen in dem Datastructure-Paket gehalten und persistent gemacht.

Klasse ArgumentException:
ArgumentException ist eine Fehlerklasse für ein falsches Übergabeelement.

Klasse CalibrationObject:
Das CalibrationObject stellt die Repräsentation des Kalibrationsgegenstandes für die Segmentierung dar.

Klasse DICOMReader:
Liest die Bilddaten ein, welche im PNG-oder DICOM-Format vorliegen können. Unterstützt werden nur Graustufenbilder mit 8, 12 oder 16 Bit Farbtiefe. Außerdem werden aus den DICOM-Dateien für das Projekt relevante Informationen ausgelesen.

Klasse GenuBitMap:
Bitmaske, wird zum Speichern von Segmenten und zum Speichern der Clippingmaske benutzt.

Klasse GenuImage:
Enthält die Bilddaten, als 8- oder 16-Bit-Array und kapselt den Zugriff auf diese nach außen hin. Die Bilddaten können als PNG-Datei gespeichert werden. Außerdem erstellt das GenuImage auch die zugehörigen Thumbnails.

Klasse GeoData:
GeoData kapselt alle für die 3D-Rekonstruktion benötigten Daten. Zu diesen gehören sowohl die zweidimensionalen Eingabekoordinaten als auch die rekonstruierten Diamantpunkte, Kamerapositionen und zusätzliche Zwischenergebnisse.

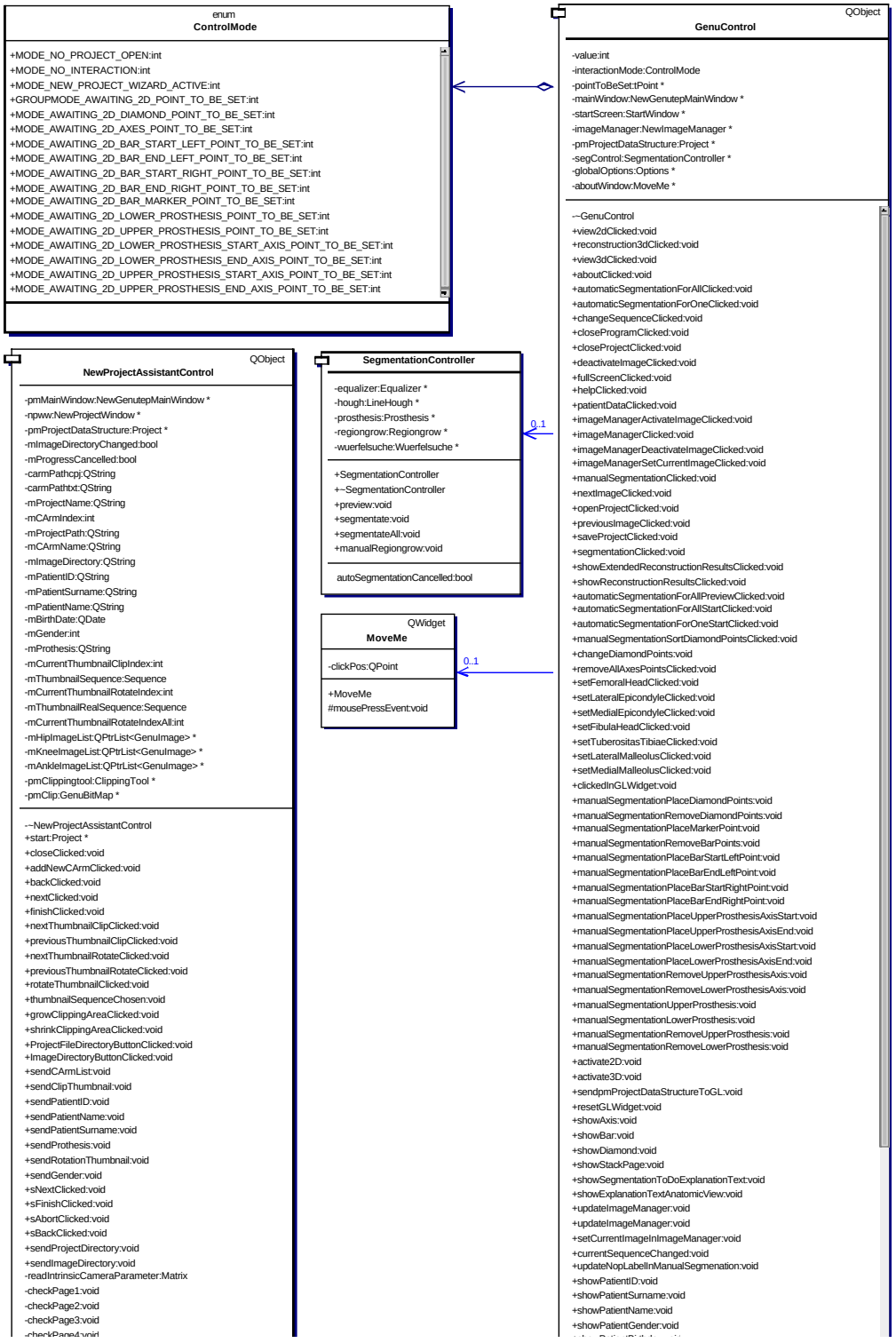


Abbildung A.4: Klassendiagramm Package Control

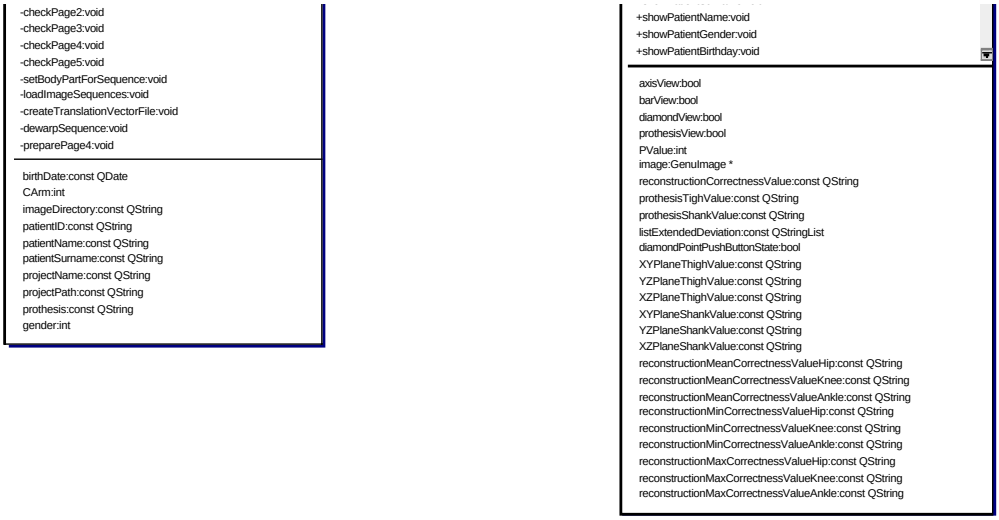


Abbildung A.5: Klassendiagramm Package Control - forts.

Klasse ImageSequence:
Enthält die Bilder einer Sequenz (Hüfte, Knie oder Knöchel) und das zu ihnen gehörige GeoData-Objekt.

Klasse Options:
Die Optionsklasse enthält alle globalen Einstellung der Anwendung.

Klasse Patient:
Enthält die entweder ausgelesenen oder die manuell eingegebenen Patientendaten, wie z.B. den Namen und den Geburtstag.

Klasse PictureDataSet:
Enthält die drei Imagesequenzen und kapselt den Zugriff auf die einzelnen Bilder nach außen hin. Falls nicht alle Bilder im Speicher gehalten werden können, lädt PictureDataSet automatisch nicht im Speicher vorhandene Bilder je nach Bedarf nach.

Klasse Project:
Enthält einen Patientendatensatz und einen Bilddatensatz.

Klasse ReadDICOMThread:
Liest die Bilddaten aus DICOM-Dateien und speichert diese in ein GenuImage.

Klasse ReadPNGThread:
Liest die Bilddaten aus PNG-Dateien und speichert diese in ein GenuImage.

Klasse ReconstructionContainer:
Speichert die rekonstruierten Punkte in einer Matrix ab, und beinhaltet eine Transformationsmatrix, die rekonstruierten Diamantenpunkte und eine Rotationsmatrix. Alle Matrizen lassen sich transformieren und zurückgeben.

Klasse SegmentationContainer:
Der SegmentationContainer stellt einen erweiterbaren Datencontainer für 2D-Segmentierungsergebnisse zur verfügung. Alle Segmente finden sich hier als private Attribute wieder.

A.3.4 Package Dewarping

Das Dewarping-Paket bietet die neben der Entzerrung der C-Bogen-Aufnahmen eine Anbindung an das externe Kalibrierungswerkzeug.

Klasse AlphaBerechnung:
Die Methode alphaberechung berechnet die Verschiebevektoren für jedes einzelne Pixel des zu dewarpten Bildes. Zum Berechnen der Verschiebevektoren wurde die Methode der radialen Basisfunktion benutzt.

Klasse ImageCenter:
Diese Klasse übernimmt die Berechnung des Referenzgitters und beinhaltet zusätzlich die Methoden zum Auslesen der XML Dateien des Kalibrierungstools.

Klasse Point:
Die Klasse dient als Hilfsklasse zur Speicherung von Objekten vom Typ Punkt inklusive der zugehörigen Referenzkoordinaten und Verschiebevektoren.

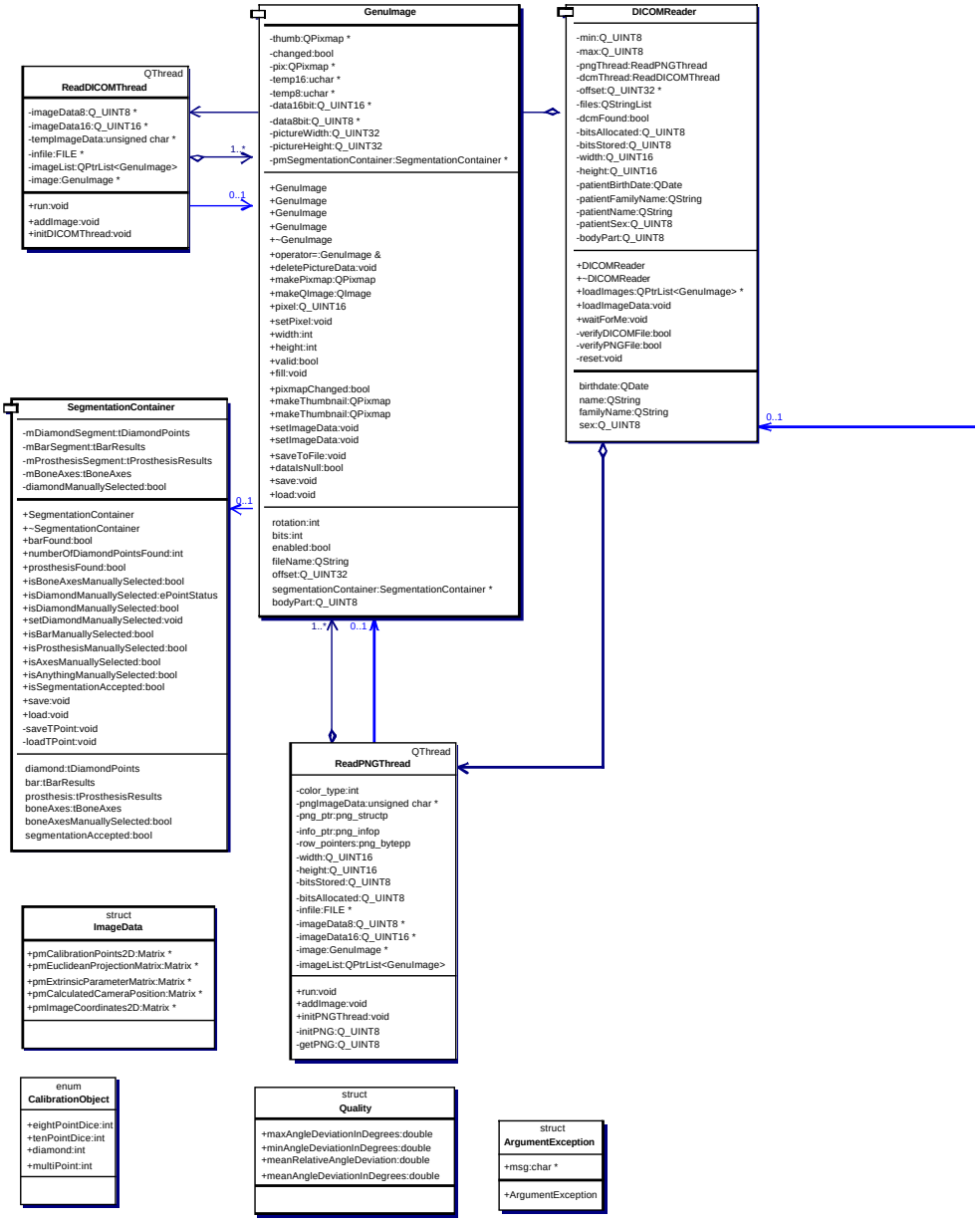


Abbildung A.6: Klassendiagramm Package Datastructure

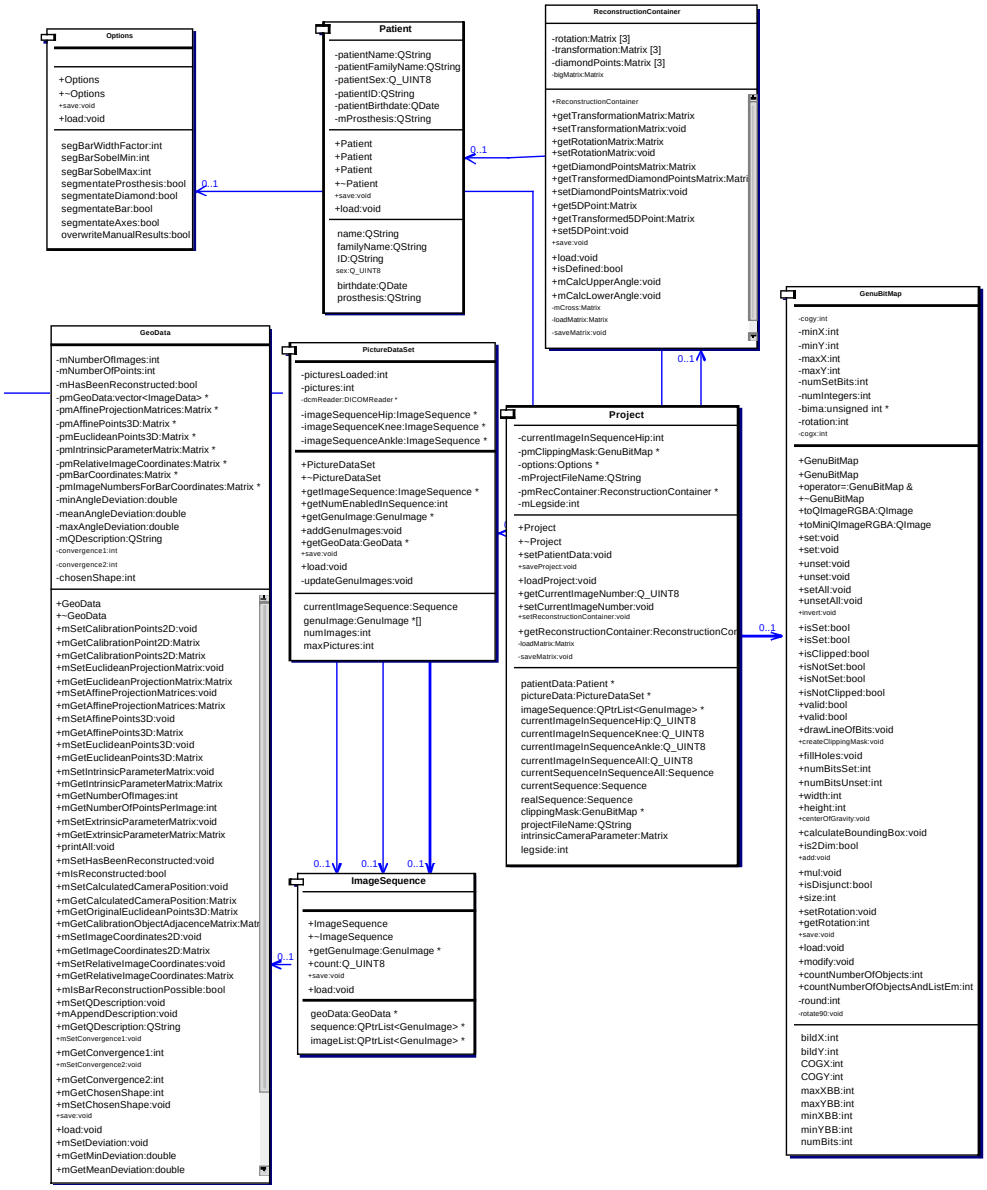


Abbildung A.7: Klassendiagramm Package Datastructure - forts.

Klasse ProgressBar:

Die Klasse diente lediglich als Prototyp um die Funktionalität des Dewarpingtools zu einem Zeitpunkt zu testen, als die komplette GUI noch nicht fertig war.

Klasse RefPoint:

In dieser Klasse findet der Aufruf der eigentlichen Dewarping Routinen statt. Ein Aufruf der Methode processStart bewirkt den Ablauf des Einlesevorgangs und die Berechnung des Referenzgitters. Anschließend erfolgt die Berechnung der Referenzpunkte und deren Zuweisung zum zweidimensionalen Vektor vom Typ Point

Klasse Vector2D:

Die Klasse Vector2D wird dazu benutzt, um die ausgelesenen Gitterkoordinaten in einen Vector zu schreiben und anschließend nach zuerst nach y und dann nach x zu sortieren. Danach stehen die Koordinaten in der richtigen Reihenfolge (x. Zeile: x.1 x.2 x.3 x.4 ...) dies entspricht dem Aufbau des Referenzgitters (Refx.1 Refx.2 ...). Daher wird zur Berechnung der Verschiebevektoren Refx.1 nach x.1 zugeordnet. Dies beruht auf der Annahme dass die Verzerrung nicht zu groß ist, was allerdings legitim sein sollte.

A.3.5 Package GUI

Dieses Paket realisiert alle Anzeigefunktion. Es sind Klassen für die Darstellung von Qt-Widgets, also auch für 2D- und 3D-OpenGl-Anzeigen enthalten.

Klasse MainWindow:

Die Klasse MainWindow beinhaltet die mit Hilfe des Qt-Designers erstellte graphische Oberfläche des Programms.

Klasse BoneLoader:

Der BoneLoader läd als separater Qt-Thread im Hintergrund einen Ober- und Unterschenkelknochen ein. Die Daten der Knochen liegen als Dreieckskoordinaten in einem proprietärem Textformat vor aus denen über den BoneLoader eine OpenGL Repräsentation erzeugt wird.

Klasse GenuGLWidget:

Die von QGLWidget abgeleitete Klasse GenuGLWidget enthält alle Funktionen für die zwei- und dreidimensionale Darstellung sowie die Routinen für die Maus- und Tastaturnavigation.

Klasse ImageManager:

Die mit dem Qt-Designer erstellte Klasse ImageManager enthält das Layout für den Bildmanager, in dem Thumbnails der einzelnen Aufnahmebereiche (Hüfte, Knie, Knöchel) angezeigt werden.

Klasse NewGenutepMainWindow:

Diese von MainWindow abgeleitete Klasse beinhaltet Methoden zur Verknüpfung der GUI-Elemente mit der Hauptkontrollklasse GenuControl.

Klasse NewImageManager:

Die von ImageManager erbende Klasse newImageManager enthält zusätzliche Signals und

Abbildung A.8: Klassendiagramm Package Dewapring

Slots, um mit der Klasse GenuControl zu kommunizieren, den Bildmanger mit Thumbnails zu befüllen und Thumbnails zu aktivieren bzw. deaktivieren.

Klasse NewProjectWindow:
Diese Klasse, die von StartWizard abgeleitet wurde, verbindet Signal- und Slot-Methoden für die Kommunikation mit der Instanz der Klasse NewProjectAssistantControl. Damit ist die Verarbeitung der in der NewProjectWindow-Instanz eingegebenen Patienten-, Bild- und Projektdaten in der Instanz von NewProjectAssistantControl möglich.

Klasse StartWizard:
Die Klasse StartWizard erbt von QWizard. Sie beinhaltet die graphische Oberfläche des Fensters zum Anlegen neuer Projekte.

A.4 Package Segmentation

Das Segmentation-Paket enthält alle angewandten Segmentierungsalgorithmen und die für die Anwendung notwendigen Filter.

Klasse ArcHough:
Diese Klasse stellt die Funktionalität der Kreis-Hough Transformation bereit (Siehe auch Kapitel 4.2.4.2).

Klasse ArcHoughSpace:
Dient als Speicherklasse für einen Kreis-Houghraum.

Klasse ClippingTool:
Das ClippingTool bietet Funktionen um eine Clippingmaske zu erzeugen. Die Maske wird über zwei Methoden in ihrer Größe manipuliert und schließlich auf jedes Eingabebild angewandt.

Klasse Equalizer:
Verstärkt ein GenuImage, so dass danach eine Gleichverteilung der Helligkeitswerte vorliegt.

Klasse Gauss:
Die Klasse Gauss stellt einen Filter für die Bildglättung dar, dabei kann die Kerngröße frei angegeben werden.

Klasse LineHough:
Stellt, basierend auf der Hough-Transformation, Funktionen zur Staberkennung bereit.

Klasse LineHoughMaxima:
Speichert eine temporäre Tabelle, in der die Maxima des zugehörigen Hough-Raumes in absteigend sortierter Reihenfolge gespeichert sind.

Klasse LineHoughSpace:
Speichert den Hough-Raum einer Linien-Hough-Transformation.

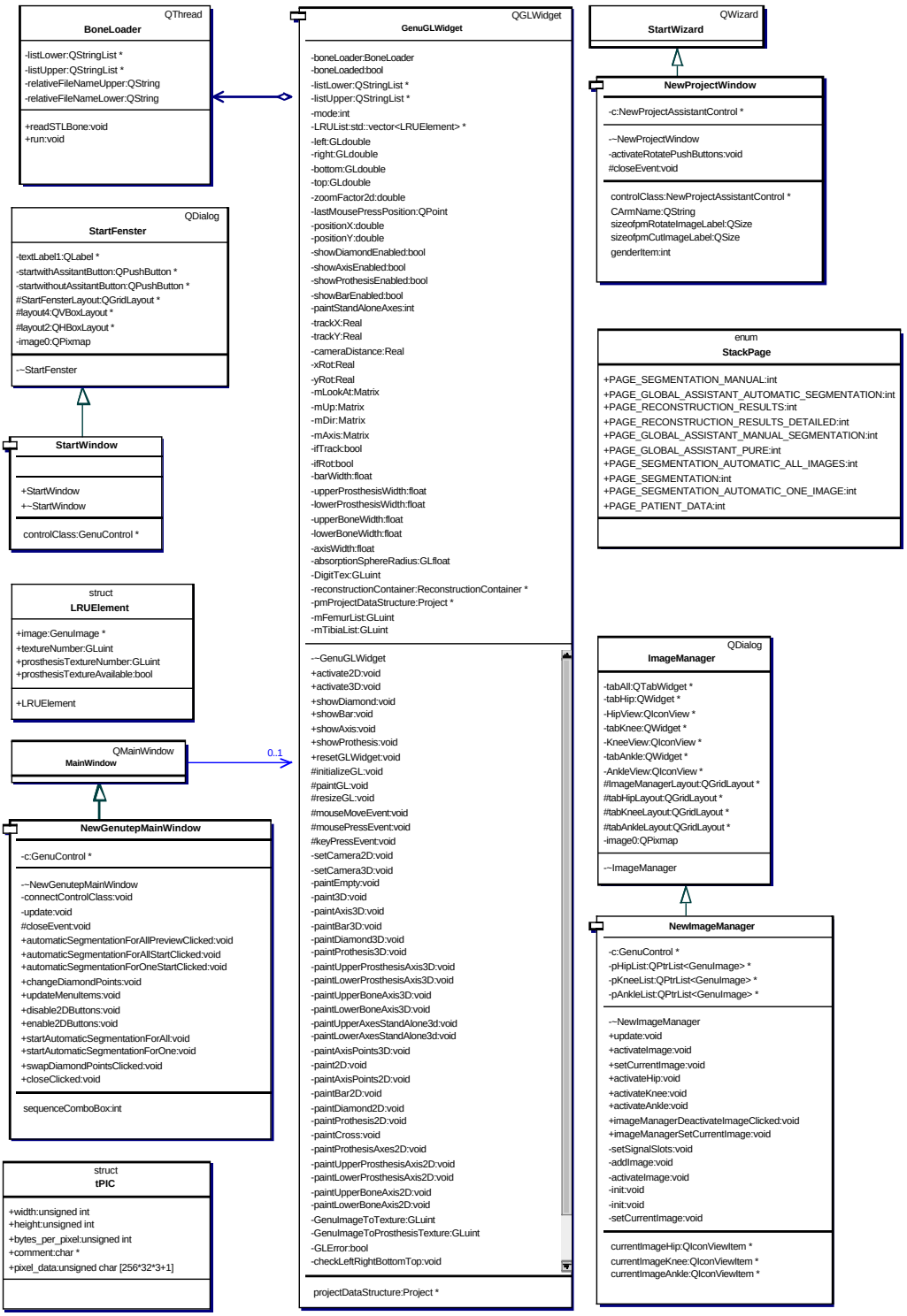


Abbildung A.9: Klassendiagramm Package GUI

Klasse Prosthesis:
Die Prosthesis-Klasse dient zur automatischen Prothesensegmentierung. Dabei sind alle Einstellungen für das Regiongrow-Verfahren, sowie die Koordinanten der manuellen Prothesenachsensegmentierung hinterlegt.

Klasse Regiongrow:
Mit der Klasse Regiongrow wird ein einfaches Bereichswachstumsverfahren durchgeführt. Die entsprechenden Eingabeparameter werden aus der Klasse Prosthesis übernommen, da dieses Verfahren nur für die automatische Prothesensegmentierung verwandt wird.

Klasse Sobel:
Eine Kantenerkennung mittels Sobel-Kantendetektor wird durch die Klasse Sobel bereitgestellt.

Klasse Threshold:
Threshold stellt einen Schwellenwertfilter für die Prothesensegmentierung bereit.

Klasse WaveletFilter:
Filtert ein GenuImage, so dass grobe Störungen entfernt werden. Dazu wird eine Wavelett Transformation angewandt, gefiltert und wieder zurücktransformiert.

Klasse Wuerfelsuche:
Diese Klasse „sucht“ die Diamantpunkte, aus „historischen“ Gründen heißt sie noch Würfelsuche. Es wird die Klasse archough benutzt, der genaue Algorithmus wird in Kapitel 4.4.4.1 beschrieben.

Klasse WuerfelZuordnung:
Mit dieser Klasse werden die gefunden Punkte in die richtige Reihenfolge gebracht, damit die Rekonstruktion eindeutige Daten erhält.

A.5 Package src

Das Standardpaket enthält neben globalen Konstanten und globalen Datenstrukturen (struct, enum, typedef) die globale Exception-Klasse.

Klasse GenuException:
Enthält Informationen über das Modul, in dem der Fehler auftrat, über die Schwere des Fehlers sowie eine textuelle Beschreibung.

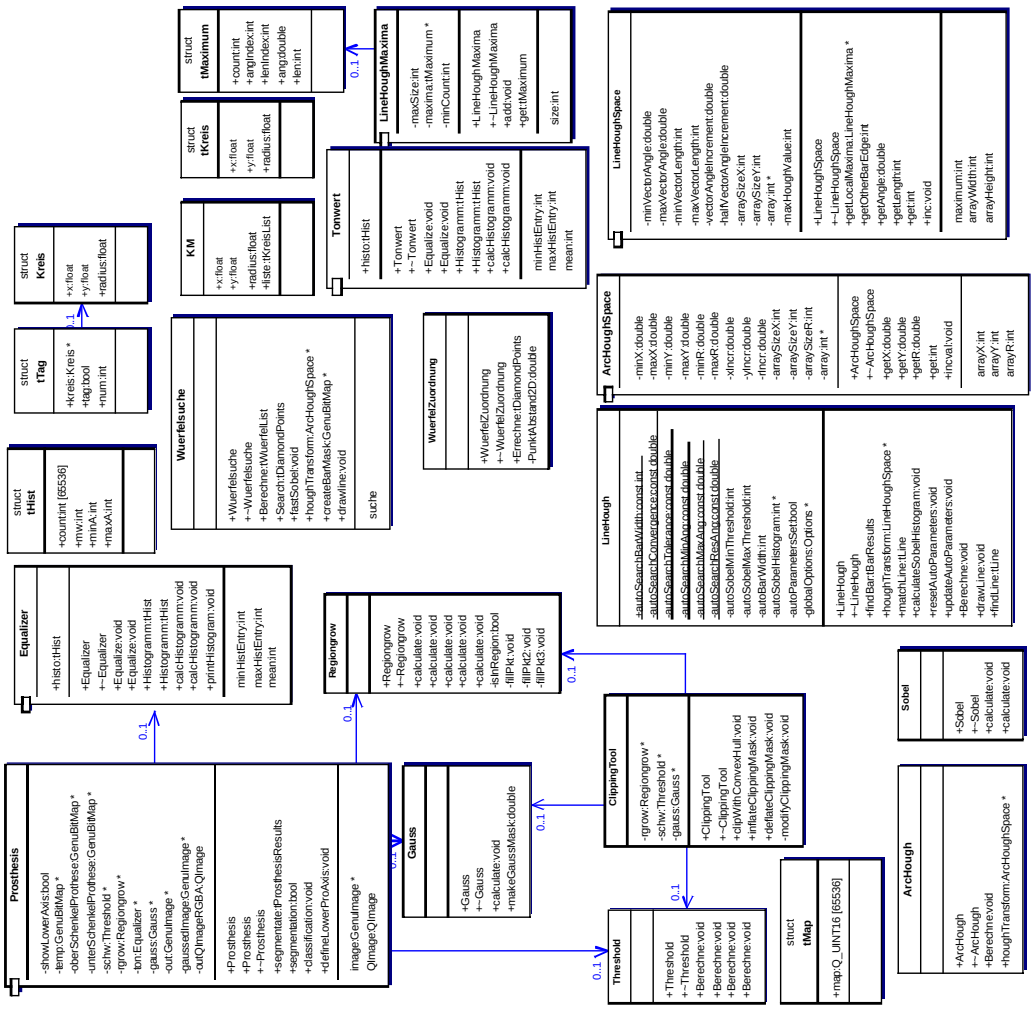


Abbildung A.10: Klassendiagramm Package Segmentation

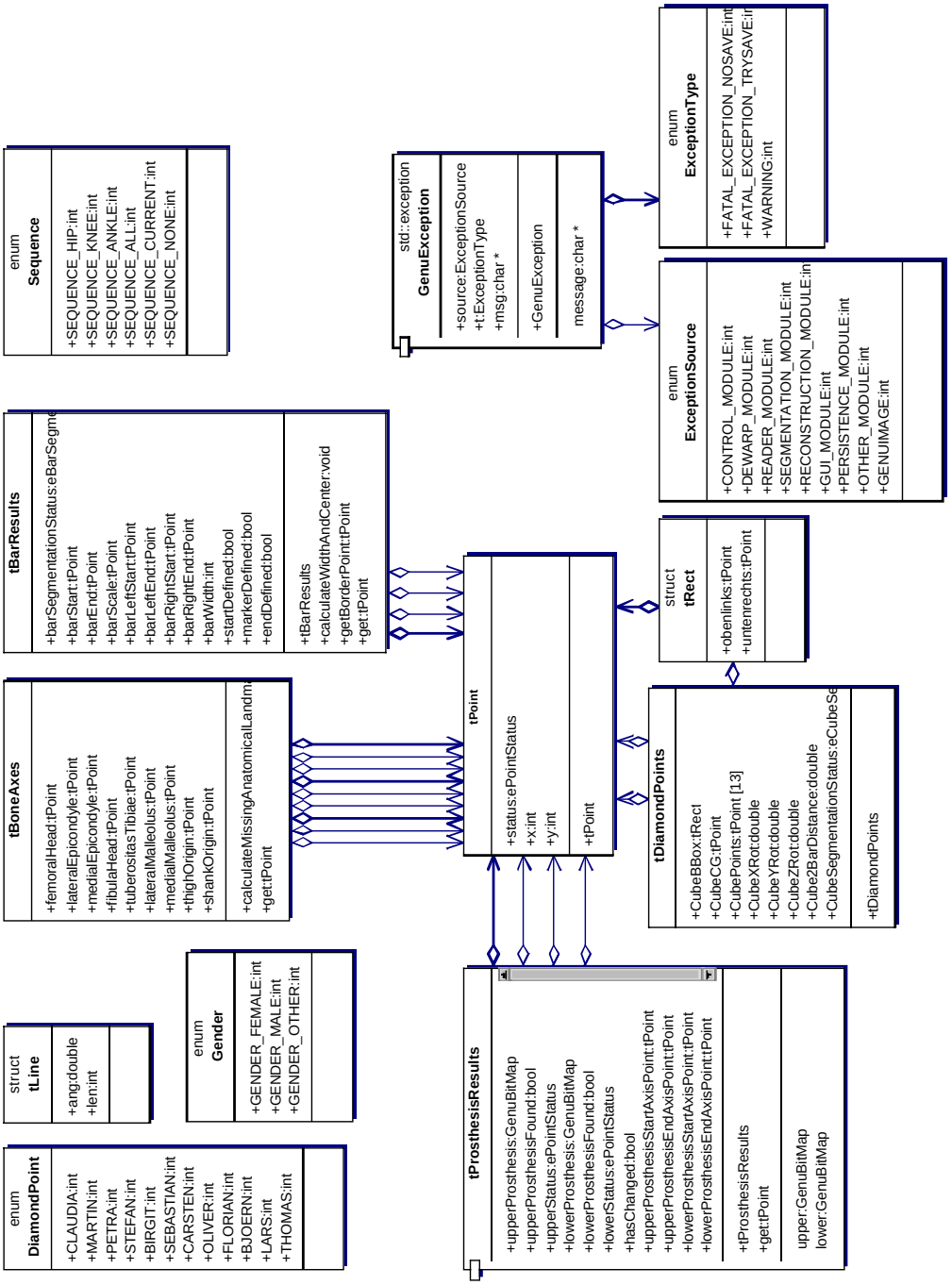


Abbildung A.11: Klassendiagramm des Defaultpaketes

Anhang B

Pflichtenheft

B.1 Zielbestimmung

Die GenuTEP-Applikation dient dem Zweck der postoperativen Lagekontrolle von Knieprothesen. Dazu werden aus einer Serie von Fluoroskopiebildern verschiedener anatomischer Bereiche (Hüfte, Knie, Knöchel) die Oberschenkel-, Prothesen- und Unterschenkelachsen dreidimensional rekonstruiert und der Winkel zwischen diesen berechnet.
Der gesamte Verarbeitungsprozess von der Datenakquisition, mit Hilfe eines C-Bogens, bis hin zur Achsenrekonstruktion und Visualisierung durch das GenuTEP-Programm wird in Abbildung B.1 dargestellt. Die GenuTEP-Applikation setzt dabei voraus, dass die Kalibrierungsaufnahmen (siehe B.1.1.1), sowie die Positionierung des Kalibierungswürfels und -stabes im Produktiveinsatz, den Anweisungen des mitgelieferten Handbuches folgen.

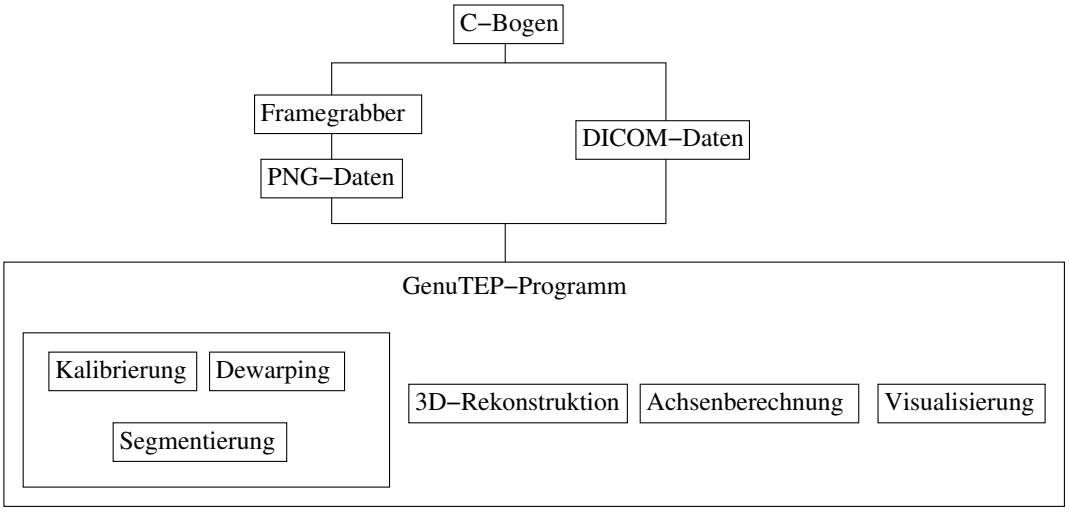


Abbildung B.1: Übergreifendes Modulschema

Der Vorgang der Achsenrekonstruktion wird durch folgendes Prozessmodell veranschaulicht:
Bevor eine Achsenrekonstruktion durchgeführt werden kann, muss eine Kalibrierung des C-Bogens durchgeführt werden:

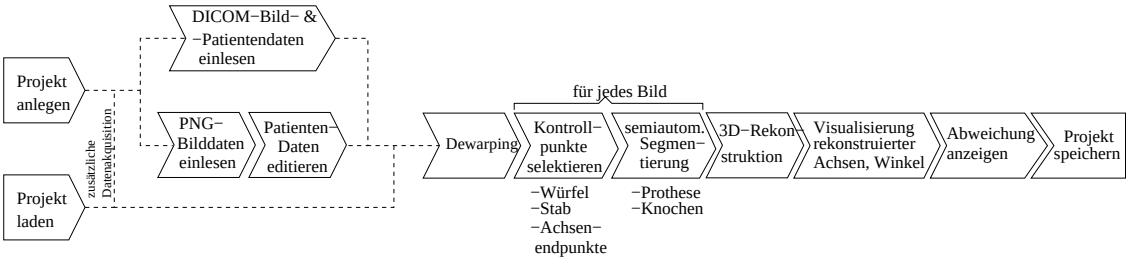


Abbildung B.2: Prozessmodell für die Achsenrekonstruktion

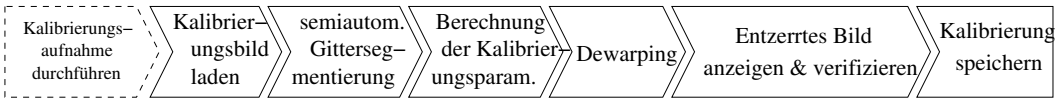


Abbildung B.3: Prozessmodell für die Kalibrierung

Die GenuTEP-Applikation basiert auf mehreren Modulen. Abbildung B.4 listet die Komponenten in ihrer Verarbeitungshierarchie auf. Dabei stellt ein Durchschnitt von oben nach unten die gesamte Verarbeitungspipeline dar. Das Projekt aggregiert alle in den Komponenten anfallenden relevanten Daten.

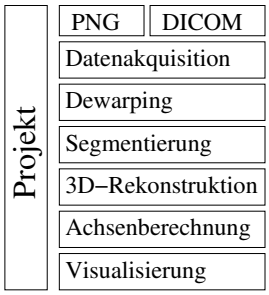


Abbildung B.4: Modulsicht

B.1.1 Musskriterien

Die im Folgenden aufgeführten Leistungen sind für das Produkt und seinen erfolgreichen Einsatz unabdingbar und müssen auf jeden Fall erfüllt werden.

B.1.1.1 Kalibrierungsfunktion

Das Kalibrierungsverfahren der GenuTEP-Applikation basiert auf der Akquise von mindestens drei Aufnahmen eines, im Lieferumfang enthaltenden, planaren Kalibrierungsgitters. Die Aufnahmen werden aus verschiedenen Aufnahmewinkeln erfasst, wobei die Gitterposition nicht verändert werden darf. Zusätzlich wird eine Aufnahme mit einem direkt vor dem Bildverstärker montierten Gitter, dass ebenfalls im Lieferumfang enthalten ist, akquiriert. Die aus diesen Aufnahmen entstandenen Kalibrierungsdaten werden dann für jeden kalibrierten C-Bogen im System gespeichert.

B.1.1.2 Laden der Bilddaten

Die Software muss in der Lage sein, Bilddaten aus dem DICOM-Format einzulesen. Dabei beschränkt sich diese Fähigkeit auf die Codierungsvariante der vorliegenden DICOM-Testdaten. Das bedeutet, es müssen keine beliebigen DICOM-Varianten unterstützt werden. Das DICOM-Format liefert 12-bit-Grauwertbilder.
Es wird möglich sein, Bilddaten in dem Dateiformat PNG einzulesen. Dabei werden 8- und 16 bit Grauwertbilder unterstützt.
Einzulesende medizinische Bilddaten beziehen sich auf Sequenzen von Fluoroskopiebildern aus den anatomischen Bereichen Hüfte, Knie und Knöchel. Des Weiteren müssen zu Kalibrierungszwecken Fluoroskopiebilder des vom Hersteller vorgegebenen Kalibrierungsgitters im entsprechenden DICOM-Format eingelesen werden.

B.1.1.3 Eingabe von Patientendaten

Im Programm ist die Mitführung der zu einem Bilddatensatz benötigten Patientendaten vorgesehen. Patientendaten sind in diesem Zusammenhang die zur Identifikation benötigten Informationen, wie zum Beispiel Name, Geburtsdatum, ID und Geschlecht sowie weitere medizinisch relevante Daten.
Die Eingabe der Patientendaten erfolgt in erster Linie durch das Auslesen der entsprechenden Tags der DICOM-Dateien. Des Weiteren soll eine manuelle Eingabe durch das medizinische Personal ermöglicht werden.

B.1.1.4 Dewarping

Vor der weiteren Verarbeitung der eingelesenen Bilddaten müssen diese zunächst entzerrt (dewarped) werden. Die für das Dewarping benötigten Bildinformationen werden aus zuvor eingelesenen Fluoroskopiebildern des Kalibrierungsgitters ermittelt. Das Kalibrierungsverfahren soll unabhängig vom Typ des verwendeten C-Bogens funktionieren.

B.1.1.5 Segmentierung

In den eingelesenen Bilddaten müssen die acht Eckpunkte des Kalibrierungswürfels, der Kalibrierungsstab sowie die Prothese und die Knochen segmentiert werden. Die Segmentierungsalgorithmen arbeiten semi-automatisch und müssen interaktiv vom Benutzer unterstützt werden.

B.1.1.6 3D-Rekonstruktion

Mithilfe der in jedem Bild ermittelten Würfeleckpunkte wird eine 3D-Rekonstruktion durchgeführt. Das Ergebnis der Rekonstruktion sind die dreidimensionalen Raumkoordinaten sowie die Ausrichtung der Kamera. Diese werden für jedes Bild ermittelt.
Voraussetzung für die 3D-Rekonstruktion ist die Sichtbarkeit aller acht Eckpunkte des vom Hersteller vorgegebenen Kalibrierungswürfels in jeder eingegebenen Fluoroskopieaufnahme. Ebenfalls zwingend erforderlich ist die Sichtbarkeit des vorgegebenen Kalibrierungsstabes in jeder Aufnahme. Dabei muss sichergestellt werden, dass die relative Position von Würfel, Stab und Patient während der gesamten Aufnahmezeit unverändert bleibt.

B.1.1.7 Achsenbestimmung

Zur Achsenberechnung für Oberschenkel, Unterschenkel und Prothese werden vom Benutzer die Achsenendpunkte in den Fluoroskopiebildern markiert. Für jeden Achsenendpunkt sind manuelle Markierungen in mindestens drei Bildern erforderlich. Das Programm errechnet daraus die Lage der Achsen im dreidimensionalen Raum und gibt den Winkel zwischen den Prothesen- und Schenkelachsen sowie eine Abschätzung der bei der Berechnung erzielten Genauigkeit an.

B.1.1.8 Achsenvisualisierung

Die Achsenvisualisierung erfolgt sowohl zweidimensional als auch dreidimensional. Die zweidimensionale Darstellung wird durch eine Einblendung der Achsen und Winkel in den entsprechenden Fluoroskopiebildern farbig realisiert. Im Dreidimensionalen wird ein rotierbares Linienmodell der Achsen und Winkel erstellt.

B.1.1.9 Teilrevidierung von Benutzereingaben

Zur Steigerung der Benutzerfreundlichkeit ist es sinnvoll, dass der Benutzer während der verschiedenen Phasen der Berechnung (Bildeingabe, Segmentierung, Achsenbestimmung) bereits durchgeführte Eingaben revidieren und korrigieren kann.

B.1.1.10 Datenspeicherung

Die von der Software errechneten geometrischen Daten, wie zum Beispiel die Lage der Achsen und der Winkel zwischen den Achsen, sowie die eingegebenen Patientendaten müssen persistent gesichert werden können.

B.1.2 Wunschkriterien

Folgende Kriterien beschreiben Wünsche an das Produkt, die nicht zwingend erfüllt werden müssen, aber angestrebt werden sollten.

B.1.2.1 Volumenvisualisierung

Wünschenswert ist die Visualisierung der Anatomie als dreidimensionales Voxelmodell. Diese Form der Visualisierung ermöglicht eine plastische Darstellung des Knochens und der darin liegenden Prothese. Mittels beliebig im 3D-Modell platzierbarer Schnittebenen könnte sich der Benutzer einen direkten räumlichen Eindruck von der Lage der Prothese verschaffen. Auch in dieser Darstellungsform können die berechneten Achsen und Winkel eingeblendet werden.

B.1.2.2 C-Bogen-Verwaltung

Es soll möglich sein, die Konfigurationsdaten mehrerer C-Bögen so abzuspeichern, dass diese bei der Bearbeitung eines neuen Bilddatensatzes – ohne eine erneute Kalibrierung – verwendet werden können.

B.1.2.3 Patientenbezogene Projekte

Patientenbezogene Projekte sollen es ermöglichen, für je einen Patienten ein “Projekt“ anzulegen, in welchem die Patientendaten und die ermittelten geometrischen Informationen für *mehrere* eingelesene Bildsequenzen gespeichert werden. Das heißt, es muss nicht für jede neue Untersuchung eine neue Datei angelegt und die Patientendaten nicht jedes Mal neu eingegeben werden. Patientenprojekte könnten auch neue Funktionen wie z.B. das Vergleichen mehrerer Achsendaten unterschiedlicher Zeitpunkte bei einem Patienten ermöglichen.

B.1.2.4 Automatische Achsenpunkterkennung

Es ist wünschenswert, dass die für die Achsenbestimmung erforderlichen Achsenendpunkte in den Fluoroskopiebildern *mit einem hohen Grad an Automatismus* ermittelt werden. Dabei ist eine Interaktion mit dem System vorgesehen, so dass Korrekturmöglichkeiten und die Endkontrolle durch den Benutzer weiterhin möglich sind.

B.1.2.5 DICOM-Speicherung

Die von der Software ermittelten geometrischen Informationen sowie die benutzten Bild- und Patientendaten werden im DICOM-Format gespeichert. Dabei wird so weit möglich auf allgemein anerkannte DICOM-Tags zur Kennzeichnung der Daten zurück gegriffen.

B.1.2.6 Framegrabbing

Es wird ein externes Framegrabbing-Modul erstellt, welches aus Videodaten, die von einem C-Bogen ausgegeben werden, Videoframes extrahiert und diese in einem für die Software geeigneten Format abspeichert. Diese Bildinformationen können anschließend als Eingabedaten für das Programm verwendet werden.

B.1.3 Abgrenzungskriterien

Die Abgrenzungskriterien beschreiben, welche Ziele mit dem Produkt bewusst nicht verfolgt werden.

B.1.3.1 Ausschließlich postoperative Kontrolle

Das Programm dient ausschließlich der postoperativen Lagekontrolle von Knieprothesen. Es soll nicht intraoperativ eingesetzt werden. Des Weiteren ist es nicht dazu geeignet, vor dem Einsetzen einer Prothese ihre optimale Einbauposition zu bestimmen.

B.1.3.2 Keine Patientendatenbank

Die Software ist und beinhaltet keine Patientendatenbank. Es werden lediglich die vom Benutzer eingegebenen Patientendaten (siehe B.1.1.3) mitgeführt. Erweiterte Funktionalitäten, wie sie in Datenbanken angeboten werden, sind nicht vorgesehen. Insbesondere gibt es keine Such-, Sortier- oder Auswertungsfunktionen.

B.1.3.3 Keine Anbindung an andere Systeme/Datenbanken

Das Produkt arbeitet als eigenständige Software und erfordert keine Anbindung oder Verwendung von bereits bestehende Softwaresysteme oder Datenbanken.

B.1.3.4 Keine beliebigen Formate

Die benötigten Bilddaten können nicht aus beliebigen (Datei-)Formaten eingelesen werden. Es werden ausschließlich Siemens SPI DICOM 3.0 und PNG als Dateiformate unterstützt.

B.2 Produkt-Einsatz

B.2.1 Anwendungsbereich

Das Produkt soll der postoperativen Lagekontrolle von Knieendoprothesen dienen. Das Produkt ist für intra- und präoperative Einsätze zur Zeit nicht vorgesehen.

B.2.2 Zielgruppen

Benutzer/Zielgruppen dieses Systems sind Ärzte und MTRA (Medizinisch-technische radiologische AssistentInnen) mit durchschnittlichen PC-Kenntnissen. Nicht geeignet ist dieses System für Personen ohne radiologisches und medizinisches Fachwissen.

B.2.3 Betriebsbedingungen

Das System mit der entsprechenden Hardware kann sowohl in der Radiologie als auch im Büro oder anderen unsterilen Räumen platziert werden. Ungeeignete Umgebungen sind OP-Säle und sonstige sterile Räume. Das System muss als dediziertes System laufen.

B.3 Produkt-Umgebung

B.3.1 Software

Als Betriebssystem kommt Linux/Windows zum Einsatz.
Zur grafischen Ausgabe wird unter Linux X-Windows (X11R6) eingesetzt, wobei auf die Qt-Bibliothek (Oberfläche) und die OpenGL-Bibliothek (3D-Visualisierung) zurückgegriffen wird.
Die verwendete Programmiersprache ist C++, um eine schnelle Verarbeitung und interaktive Benutzung zu ermöglichen. Qt soll in der aktuellen Version 3.x eingesetzt werden. OpenGL wird in Version 1.3 zum Einsatz kommen. Unter Linux ist die glibc ab Version 2 als zu verwendende C-Bibliothek vorausgesetzt.
MS-Windowsbasierte Betriebssysteme werden in den Versionen Windows 2000 Professional und Windows XP (Pro, Home) unterstützt.

B.3.2 Hardware

Das Programm läuft auf einem PC mit folgenden Hardware-Mindestanforderungen:

1. Intel Pentium 4 (2.0 GHz)

2. 1 GB RAM
3. Grafikkarte: nVidia GeForce 4 (256 MB)

Alle akquirierten Bilddaten werden aus festinstallierten oder mobilen C-Bögen extrahiert.

Zur Digitalisierung der Daten aus einem videokompatiblen C-Bogen wird eine Framegrabber-Karte vorausgesetzt. Eine Linux-Library zur Ansteuerung der Karte sollte vorhanden sein.

Zur Entzerrung der Bilder (Dewarping) wird ein planares Kalibrierungsgitter benutzt. Weitere Kalibrierinstrumente sind ein Kalibrierwürfel und ein Kalibrierstab. Die Kalibriergeräte werden vom Hersteller vorgegeben.

B.4 Produkt-Funktion

Im Folgenden werden die Funktionsbausteine der GenuTEP-Applikation aufgelistet. Funktionen sind mit **/F*Funktionsnummer*/** eindeutig bezeichnet. Ein nachgestelltes *W* kennzeichnet ein Wunschkriterium.

B.4.1 Projektverwaltung

/F10/ Projekt anlegen
Ein neues Projekt wird eingabeassistentengestützt erstellt. Dabei wird ein Projektname als Eingabe erwartet. Ein Projekt bildet den Rahmen für alle in den weiteren Prozessen akquirierten und errechneten Daten.

/F20/ Patientendaten neu eingeben
Patientendaten müssen manuell eingegeben werden, falls die erforderlichen Informationen nicht aus einer DICOM-Datei eingelesen werden konnten oder als Eingabe das PNG-Format gewählt wurde.
Es werden folgende Daten erfasst: Name, Vorname, Geburtsdatum, Geschlecht, Patientenreferenznummer.

/F30/ Patientendaten ändern
Es besteht die Möglichkeit, Patientendaten in jedem Prozessschritt menügesteuert zu ändern. Diese Änderungen propagieren sich nicht in die DICOM-Dateien der Eingabe.

/F40/ Projekt laden
Ein bereits existierendes Projekt kann mithilfe eines Dateiauswahldialoges geladen werden.

/F50 W/ C-Bogen auswählen
Aus einer Liste von bereits kalibrierten C-Bögen kann der gewünschte C-Bogen ausgewählt werden. Die entsprechenden Kalibrierungsinformationen werden global eingestellt.

/F60/ Bilddaten wählen
Die Daten werden im Siemens SPI DICOM 3.0 Format (12 bit) bzw. als Grauwert-PNG (8- oder 16 bit) eingelesen. Falls PNG als Eingabeformat dient, wird eine eindeutige Zuordnung von Aufnahmereihenfolge und anatomischer Struktur zu den Dateinamen erwartet.

/F70 W/ Bilddaten löschen
Fehlerbehaftete oder für den Bearbeitungsprozess nicht mehr benötigte Aufnahmen können aus dem Projekt gelöscht werden.

/F80/ Projekt speichern
Die Projektdaten werden für einen eventuell späteren Zugriff gespeichert. Es werden Patientendaten, C-Bogenauswahl, Segmentierungsinformationen, Ergebnisse der 3D-Rekonstruktion sowie berechnete Achsen und Winkel in einem proprietären Format gesichert.

/F90 W/ Export
Neben dem obigen Standard-Dateiformat können die Projektinformation in XML und einem eigenen DICOM-kompatiblen Format abgelegt werden.

B.4.2 Globale Einstellungen

/F100/ Berechnung intrinsischer Parameter
Nach Einlesen des Kalibrierungsbildes werden die intrinsischen Parameter berechnet. Die Positionen der Kalibrierungspunkte werden bestimmt und angezeigt. Anschließend erhält der Benutzer die Möglichkeit, das Ergebnis zu korrigieren. Anhand dieser Informationen werden die mit diesem C-Bogen erfassten Fluoroskopieaufnahmen entzerrt.

/F110 W/ C-Bogen-Templates speichern
Um den Betrieb mehrerer C-Bögen zu ermöglichen, werden die Kalibirierungsdaten der entsprechenden C-Bögen gespeichert.

/F120 W/ Prothesen-Templates speichern
Die Geometriedaten verschiedener Prothesen können gespeichert werden, um die semiautomatischen Segmentierungsalgorithmen zu unterstützen.

B.4.3 Achsenrekonstruktion

/F130/ Bildentzerrung
Die eingelesenen Fluoroskopieaufnahmen werden ohne explizite Benutzerinteraktion mit Hilfe von Dewarpingalgorithmen, welche die für den C-Bogen errechneten intrinsischen Parameter verwenden, entzerrt.

/F140/ Unterstützung der Segmentierung
Der Benutzer muss den semiautomatischen Segmentierungsvorgang unterstützen. Dabei werden durch Eingabe mit der Maus die interessierenden Bildbereiche markiert bzw. die Segmentierung der Würfeleckpunkte korrigiert.

/F150/ Zwischen Bildern wechseln
Es besteht die Möglichkeit, während des Segmentierungsprozesses zwischen den einzelnen Bildern zu wechseln.

/F160 W/ Anzeige von Volumendaten
Die Volumendaten werden als ausgeleuchtetes Oberflächen-Voxelmodell angezeigt.

/F170/ Achsen-Anzeige (2D)
Die Achsen werden in den 2D-Bildern als farbige 2D-Koordinatenlinien eingezeichnet.

/F180/ Achsen-Anzeige (3D)
Die Achsen werden als Linienmodell im 3D-Raum angezeigt. Die Darstellung ist rotier- und zoombar.

/F190 W/ Achsen-Anzeige (3D)
Die Achsen werden im 3D-Voxelmodell als farbige 3D-Koordinatenlinien angezeigt.

/F200/ Anzeige der Winkeldaten und -genauigkeit
Die Genauigkeit des Ergebnis wird textuell angezeigt.

/F210 W/ Navigation in der 3D-Darstellung des Voxelmodells
Es besteht die Möglichkeit die Bilder zu rotieren und in den Bildern zu zoomen.

B.5 Produktdaten

B.5.1 Bilddaten

Die zu verarbeitenden Aufnahmen umfassen die drei anatomischen Bereiche Hüfte, Knie und Knöchel. Sie liegen im DICOM- bzw. PNG-Format vor.

B.5.2 Kalibrierungsdaten

Die aus den Fluoroskopiebildern des Kalibrierungsgitters gewonnenen Daten, die zum Dewarping der Fluoroskopiebilder benötigt werden, sind zu speichern.

B.5.3 Patientendaten

An Patientendaten sind Name, Geburtsdatum, Geschlecht und Patientenreferenznummer zu speichern.

B.5.4 Achsengeometrie

Die rekonstruierten Achsen von Femur, Tibia und der Prothesenteile sowie ihre Winkel werden in den Projektdaten abgelegt.

B.5.5 Volumendaten

Die aus den Fluoroskopiebildern berechnete Segmentierung, sowie die berechnete Kameraposition und Orientierung der einzelnen Bilder werden mit in den Projektdaten gespeichert.

B.5.6 Prothesengeometrie

Die geometrischen Daten der Prothese werden persistent gespeichert, um als Prothesentemplate, wie oben erwähnt, die Segmentierung zu unterstützen.

B.5.7 Modelldaten

Neben den berechneten Daten der Achsengeometrie werden die Zwischenergebnisse der Segmentierung und 3D-Rekonstruktion gesichert.

B.6 Produktleistung

B.6.1 Winkelgenauigkeit

Die berechneten Winkel sollen idealerweise eine maximale Abweichung von 1° aufweisen. Als Fehlermaß werden sowohl numerische Ungenauigkeiten als auch entstehende Fehler bei der Kalibrierung und Segmentierung einbezogen. Der Hauptfehleranteil wird allerdings durch Ungenauigkeiten der Aufnahmen determiniert, welcher nicht vollständig heraus kalibriert werden kann.

B.6.2 Datensatz

Ein typischer Datensatz umfasst 15 bis 20 Bilder. Hierbei müssen pro anatomischer Struktur (Hüfte, Knie, Knöchel) mindestens 3 Bilder vorliegen, um eine Messung zu ermöglichen.

B.6.3 Bildgröße

Die maximal zulässige Größe eines Bildes beträgt 1024 x 1024 Pixel.

B.6.4 Maximale Volumengröße

Die maximal zulässige Größe des Volumens beträgt 512x512x512 Voxel.

B.6.5 Framerate

Die minimale Framerate bei der 3D-Volumenvisualisierung soll 3 Frames pro Sekunde betragen.

B.6.6 Bearbeitungszeit

Die Gesamtzeit eines kompletten Durchlaufes inklusive der durch den Benutzer assistierten Segmentierung sollte maximal 15 Minuten betragen.

B.7 Benutzungsoberfläche

Als Standard wird eine menüorientierte Bedienung eingesetzt.
Da man davon ausgehen muss, dass sich die Gruppe der Benutzer nicht nur aus Experten mit Computererfahrung zusammensetzt, wird eine grafische, menüorientierte und vor allem eine intuitiv bedienbare Benutzeroberfläche eingesetzt.
Als weiterer Standard ist die Bedienung mit der Maus anzusehen, welche eine zusätzliche Erleichterung darstellt.

B.8 Qualitäts-Zielbestimmungen

Die dargestellte Tabelle definiert die Schwerpunkte der Entwicklungsarbeit. Die Entwicklergruppe wird weitere Qualitätsrichtlinien aufstellen, welche die Qualität des Endproduktes sichern und die Einhaltung der oben festgelegten Kriterien überwachen werden. Dazu wird vom Projektmanagement ein angemessenes QM-Handbuch gepflegt, welches Dokumentation, Testfälle und die

Produktqualität	sehr gut	gut	normal
Präzision	x		
Portierbarkeit		x	
Benutzungsergonomie		x	
Prozessergonomie	x		
Erweiterbarkeit			x
Bedienbarkeit		x	
DICOM-Konformität			x
Keine Verfälschung der Rohdaten	x		

Tabelle B.1: Qualität des Produktes

Modellierungsgrundsätze festlegt. Autarke Modultests werden die korrekte Funktionalität der Komponenten sichern.

B.9 Globale Testfälle

B.9.1 Prüfung anhand eines Phantoms

Zur Überprüfung der Achsenberechnung wird ein Phantom erstellt. Die anhand der Aufnahmen dieses Modells berechneten Daten werden mit den realen Daten verglichen.

B.9.2 Prüfung mit zur Verfügung stehender Testdaten

Mit den Daten realer Patienten werden mehrere Testdurchläufe durchgeführt.

B.10 Entwicklungsumgebung

Die benutzte Software beinhaltet Trolltechs Qt 3.x, KDevelop 3.x der kdevelop.org und weitere Werkzeuge zur Softwareentwicklung wie CVS und LaTeX. Das Entwicklungssystem arbeitet unter Linux. Es wird ISO-C++ als Programmiersprache verwandt. Der Einsatz des gcc 3.x Compiler wird die Portierbarkeit des Produktes auf Windowssysteme gewährleisten. Des Weiteren wird Together/C++ zur Modellierung in UML verwendet.

B.11 Ergänzungen

Die Voraussetzungen zum Erhalt einer CE-Zertifizierung sollte angestrebt werden.

Anhang C

Qualitätsmanagement

C.1 Über dieses Dokument

Das vorliegende Qualitätsmanagementhandbuch soll Richtlinien und Verfahrensvorgänge beschreiben, deren Ziel es ist die Endanwendung der GenuTEP-Applikation so sicher und funktionsfähig wie möglich zu machen.
Einerseits wird die Qualität des Produktes durch ein klares, definiertes Vorgehen erreicht, andererseits, sollen Tests und strukturierte Kontrollmechanismen die korrekte, erwartete Funktionsweise sichern.
Die Inhalte des QM-Handbuches basieren einerseits auf Erfahrungen in anderen Softwareprojekten, wie auch auf Auszügen der angegebenen Literatur. Weitere Regeln und Vorgehensweisen wurden in der Diskussion der PG-Teilnehmer erarbeitet.

C.2 Allgemeine Vorgehensweise

Die Softwareentwicklung in der PG 434 GenuTEP verfolgt die zwei folgenden Paradigmen.

C.2.1 Prototypenmodell

Im ersten Entwicklungsschritt wurde das Prototypenmodell [PG 04] verfolgt. In den einzelnen Arbeitsgruppen wurden Prototypen der einzelnen Funktionsmodule erstellt. Diese Phase diente zur Evaluierung von verschiedenen Ansätzen. Die Prototypen werden nach erfolgreichen Test als Module der weiteren Entwicklung zur Verfügung stehen.

C.2.2 Wasserfallmodell

In der darauf folgenden Phase wird ein an dem Vorgehensmodell der OMG orientierter Ansatz verfolgt. Es wird das Wasserfallmodell [PG 04] angewandt. Module werden in Pakete gegliedert und einzelne Funktionen in entsprechende Klassen ausgegliedert. Es werden Klassendiagramme für sämtliche Module, Pakete und Klassen erstellt. Aktivitätsdiagramme werden für den groben Funktionsablauf definiert. Kollaborationsdiagramme sollten erstellt werden, um die Abhängigkeit und Aufrufrichtung der einzelnen Module darzustellen. Für die Sicht der Endanwenderseite werden Sequenzdiagramme, der einzelnen Aktivitäten erstellt.

C.2.3 Programmierprozess

Der Implementierungsschritt in der Prototypenphase unterliegt keiner genauen Regel. Es werden hier nur Richtlinien für die Formatierung des Quellcodes vorgegeben und eine Richtline für die Dokumentation erstellt.
In Phase Zwei wird ein XP-Ansatz vollzogen. Klassen werden immer von Arbeitsgruppen mit zwei oder drei Personen erstellt. Die Entwickler schreiben in unterschiedlichen Sitzungen abwechselnd den Quelltext, während jeweils die beisitzende Person - im gespräch mit dem Entwickler - das Vorgehen überprüft. Syntaktische Fehler werden ebenso schnell herausgefiltert, wie auch logische Denkfehler. Das Gespräch zwischen Entwickler und Auditor zwingt über die Programmflusslogik zu reflektieren. Eventuell neu anfallende Aufgaben können vom Beisitzer protokolliert werden, um sie anschließend abzuarbeiten.

C.3 Richtlinien für den Quellcode

Im folgenden werden Richtlinien für den Quelltext aufgestellt. Sie ermöglichen einen schnellen Einstieg in den Code - auch für Personen, die ihn nicht entwickelt haben. Damit soll die Wartabkeit des Quellcodes erhöht werden und typische Fehler (bspw. Pointerfehler) durch erkennbare Variablennamen ersparen.

C.3.1 Sprache

Der Quellcode wird in **englisch** geschrieben (Methodennamen, Attributnamen, etc.), die Kommentierung erfolgt in **deutsch**, wobei Fachbegriffe in englisch bleiben können.

C.3.2 Namensgebung

Die Namensgebung bei Attributen und Methoden soll sich nach der ungarischen Notation richten. Alle Bezeichnungen bestehen aus einem im Folgenden beschriebenen Präfix gefolgt von dem eigentlichen Namen. Besteht dieser aus mehreren zusammengesetzten Worten, werden jeweils die Anfangsbuchstaben der Worte groß geschrieben und der Rest klein.
Zu verwendende Präfixe:

- typedefs : t
- Pointer : p
- Member : m
- Memberpointer : pm

C.3.2.1 Classes

Klassennamen beginnen in der Definition mit einem Großbuchstaben.

C.3.2.2 Structs

C-Structs beginnen in der Definition mit einem Großbuchstaben. Wird das struct als Typdef deklariert, wird ein kleingeschriebenes t dem ersten Großbuchstaben voran gestellt.

C.3.2.3 Variablen

Instanzen von Klassen, structs, etc. beginnen mit einem kleinen Buchstaben. Wird wie oben beschrieben ein Präfix angewandt wird der erste Buchstabe nach dem Präfix groß geschrieben.

C.3.2.4 Typedefs

Typedefs beginnen mit einem Großbuchstaben mit vorangestelltem kleinem t.

C.3.2.5 Enumerations

Enum-Einträge werden nur in Großbuchstaben geschrieben.

C.3.2.6 Konstanten

Konstanten werden nur in Großbuchstaben geschrieben.

C.3.2.7 Header- und CPP-Dateien

Header- und CPP-Dateien sind klein zu schreiben.

C.3.3 Dokumentation

Die Dokumentation des Quellcodes soll mit dem Tool KDoc erstellt werden. Hierzu müssen Kommentare im bekannten Javadoc - Stil in die Headerdateien eingefügt werden. Kommentiert werden sollen alle Klassen, Methoden und Attribute.

C.3.3.1 Klassen

Jede Klasse wird mit einem KDoc-Block [Kan99] begonnen, welcher eine ausführliche Beschreibung der Klasse, den Tag *@short* mit einer Beschreibung der Klasse in einem Satz sowie die Tags *@author* und *@version* enthält:

/**
* Ausführliche Beschreibung
*
* @short <Ein Satz>
* @author <Autoren>
* @version <Version>
*/

Die Auführliche Beschreibung der Klasse wird als erstes in die Klasse eingepflegt. Sie soll einen klaren Text enthalten, welche Funktionalität diese Klasse haben soll. Es soll festgehalten werden, welche Funktionalität sich der Entwickler am Anfang von der Klasse erwartet. Der Kommentar soll deshalb absichtlich nicht am Ende der Entwicklung gepflegt werden, um das funktionale Ziel nicht aus den Augen zu verlieren.

C.3.3.2 Methoden

Vor jeder Methode steht der Methoden-Kommentar. Der Kommentar beginnt linksbündig mit der Methode und enthält eine ausführliche Beschreibung der Funktionalität der Methode sowie eine genaue Erklärung aller Parameter, Rückgabewerte und Exceptions, sofern diese vorhanden sind. Zur korrekten Dokumentation werden die Tags

- @param <parametername> <Beschreibung>
- @return<Beschreibung>
- @exception <Beschreibung>

verwendet.

C.3.3.3 Attribute

Vor jedem Attribut steht ein Kommentar, welcher die Bedeutung des Attributes erklärt.

C.3.3.4 Nicht-KDoc-Kommentare

Implementierte Algorithmen sollen zur besseren Nachvollziehbarkeit und zur Erleichterung von Tests und der Durchführung von Korrekturen angemessen kommentiert werden.
Derartige Kommentare im Quellcode beginnen stets bündig mit der Einrückung der kommentierten Zeile und stehen unmittelbar über (vor) der zu kommentierenden Stelle. Einzelige Kommentare beginnen mit // während mehrzeilige Kommentare auch mit /∗ beginnen und mit ∗/ enden dürfen.

C.3.4 Code-Formatierung

Im folgenden die Richtlinien zur Code-Formatierung:

1. Die Einrückung erfolgt mittels Tabulatoren, nicht mit Leerzeichen. Die Einrückungstiefe wird auf *vier Zeichen* festgelegt.
2. Block-Klammern ’{’ und ’}’ stehen immer in einer eigenen Zeile.
3. Zwischen zwei Methodendefinitionen werden mindestens zwei Leerzeilen eingefügt.
4. In der Headerdatei erfolgen die Deklarationen in der Reihenfolge **public**, **private**, **protected** und es werden in jedem Abschnitt zuerst Konstruktor/Destrutor, dann die Methoden und als letztes die Attribute deklariert. Zwischen zwei Abschnitten wird eine Leerzeile eingefügt.

C.3.5 Copyright

In Jeder Klasse wird als erste Zeile ein Copyright eingefügt:

„// Copyright by PG434-GenuTEP (Uni Dortmund)“

C.4 Programmfluss

C.4.1 Prüfung der Übergabeparameter

Die GenuTEP-Applikation besteht aus vielen verschiedenen Modulen, welche von der Hauptkontrollinstanz aufgerufen werden. Gewisse Module benutzen auch direkt andere Module oder Klassen. Folgende Richtlinien sollen die klassenübergreifenden Methodenaufrufe sichern.
Methoden die als öffentliche Schnittstelle zur Verfügung stehen müssen alle übergebenen Parameter überprüfen. Es sind Überprüfungen auf NULL durchzuführen. Zusätzlich sind die Übergabewerte darauf hin zu prüfen, ob sie in einem logisch korrekten Wertebereich liegen (bswp. Arraygrößen größer-gleich Null, Bildbreite nicht negativ).
Es sei kurz angemerkt, dass dies nur für öffentliche Methoden notwendig ist. Private Methoden brauchen diese Überprüfung nicht durchführen.

C.4.2 Ausnahmebehandlung

Alle Funktionen werden im ersten Schritt so durchexerziert, als ob keine Fehler auftreten können. Es werden keine überprüfende, verschachtelte if-else-Blöcke geschrieben. Es werden vielmehr im zweiten Schritt Variablen oder Prozedurergebnisse auf Schlüssigkeit überprüft und im negativen Fall eine Exception geworfen.

C.4.2.1 Systemexceptions

Tritt ein Programmflussfehler in Zusammenhang mit einer Systemressource auf, wird eine Systemexception (SystemException) geworfen. Dies ist vor allem im Modul der Dateneingabe der Fall.

C.4.2.2 Applicationexceptions

Module, die auf eigenen, oder übergebenen Datenstrukturen arbeiten, werfen Applicationsexceptions, wenn ein Fehler aufgrund von mangelhaften Übergaben (IllegalArgumentException) oder aufgrund von mangelhafter Verarbeitung durch emplementierte Algorithmen (ApplicatonException) auftritt.

C.5 Tests

Um die Zuversicht zu erhöhen, dass sich ein Programm so verhält, wie es nach der Spezifikation geplant ist, kann durch ausgiebiges Testen in verschiedenen Anwendungsszenarien erlangt werden. Es werden konstruktive und analytische Tests durchgeführt.

C.5.1 Unit Test

In Unit-Tests, als spezifikationsorientierter Test, werden in einzelnen Modulen die Funktionen überprüft und dabei Solldatum der Rückgabe mit dem tatsächlichen Rückgabewert als Istdatum verglichen. Es werden nur Methoden einem Unit-Test unterzogen, um dessen Funktionalität bezüglich des Programmflusses zu testen (Überprüfung der Übergabe, Exception-Handling). Die korrekte Funktionsweise von Algorithmen der GenuTEP-Applikation wird nur selten mit Unit-Tests möglich sein.

Als Testing-Framework wird CPPUnit (http://cppunit.sourceforge.net) eingesetzt. Es werden Tests entsprechend den CPPUnit-Vorgaben erstellt und protokolliert.

C.5.1.1 Testdaten

Der Soll- und Istdatenvergleich kann nur objektiv durchgeführt werden, wenn der Sollwert bekannt ist. Dies ist nur bei künstliche Daten möglich.
Somit werden für Unit-Test Äquivalenzklassen der Eingaben gebildet und diese als Eingabe der Unit-Tests genutzt.
Die Testdaten des Rekonstruktionsmoduls sind künstlich generierte Daten. Das Datenaquisitionsmodul und Segmentierungsmodul arbeitet auf den Aufnahmen des Phantoms. Für das Kalibrationsmodul wurden ebenfalls Testaufnahmen als Eingabe erstellt.

C.5.2 Code Review

Neben Unit-Tests, die vor allem fehlende Überprüfungen der Übergaben aufdecken, wird auch ein Code Review durchgeführt, um logische, syntaktische und allgemeine Fehler zu beseitigen. Dabei wird der Review mit Hilfe des Codereview-Formulars protokolliert.
Der Quelltext wird einem 2-Mann-Audit unterzogen. Dabei besteht das Team aus einem Entwickler des Codes und einer unbeteiligten Person. Für jede Prozedur wird ein Walkthrough durchgeführt.
Zusätzlich wird auf die Einhaltung der Codekonventionen und Dokumentationsvorgaben geachtet.

C.5.3 Funktionstest

Um die Funktionalität der Module bewerten zu können, werden die Methoden einem visuellen Test unterzogen. Dabei werden strukturiert einzelne Methoden eines Moduls oder einer Klasse aufgerufen und deren Ausgabe visuell durch das Testteam überprüft.
Für den Funktionstest werden, ähnlich der Unit-Tests, CPP-Tests aufgesetzt. Nur werden hier keine Soll-Istvergleiche durch dsa Framework vorgenommen.

C.5.4 Integrationstest

Zur Überprüfung der Gesamtfunktionalität wird ein Integrationstest nach Zusammenführen aller Module durchgeführt.

C.5.5 Anforderungsdefinitionen

In der Entwicklung von Sofwareprodukten kommen gerade in den frühen Phasen neue Anforderungen an Funktion und Schnittstellen von Modulen. Die GenuTEP-Teilnehmer stellen neue Anforderungen mithilfe von Funktionsanforderungsbögen, wie sie hier abgdruckt sind.
Der Funktionsanforderungsbogen soll schriftlich festhalten, wie oder inwiefern ein Modul verändert oder angepasst werden soll. Zusätzlich dient er als Protokoll für eventuelle Nachfragen.

Code Review-Nr.:

Modul:	Paket:	Klasse:	Version:
Team[1]:			
Codeconventions[2]: ☐	KDoc[3]: ☐	Codekommentare[4]: ☐	
Funktionsbeschreibung[5]:			
Bemerkungen[6]:			

[1] Das ausführende Testteam, 2 Personen
[2] Wurden die Spezifikation für die Quelltextformatierung, und –namensgebung eingehalten
[3] Wurden KDoc-kompatible Kommente der Methoden-, Klassen-, Strukturdefinitionen in der Header- und CPP-Datei vorgenommen
[4] Wurden im Quelltext Codepassagen adäquat kommentiert
[5] Die zu erwartende Funktion beschreiben
[6] Testergebnisse, Fehler, etc.

Abbildung C.1: Codereview

Testbogen-Nr.:

Modul:	Paket:	Klasse:	Version:

Methode:

Parameter:

Team[1]:

Funktionsbeschreibung[2]:

Vorbedingungen[3]:

Testbemerkungen[4]:

[1] Das ausführende Testteam, 2 Personen
[2] Die zu erwartende Funktion beschreiben
[3] Vorbedingungen für die korrekte Arbeit der Methode sind aufzuzeigen
[4] Testergebnisse, Fehler, etc.

Abbildung C.2: Testbogen

Funktionsanforderungsbogen-Nr.:

Modul:	Paket:	Klasse:	Version:

Methode[1]:

Parameter[2]:

Anforderung von[3]:

Priorität[4]: Datum[5]:

Funktionsbeschreibung[6]:

Bemerkungen[7]:

[1] Angaben definieren die Schnittstelle des Moduls nach außen
[2] Angaben definieren die Schnittstelle des Moduls nach außen
[3] Person, Modul, etc., welche die Funktion anfordert
[4] Priorität von 0 (gering) bis 10 (eilt)
[5] Erwartetes Datum, bis Anforderung implementiert ist
[6] Die zu erwartende Funktion beschreiben
[7] Bemerkungen (der Anfordernden Person, dem Besitzer der Klasse, etc.)

Abbildung C.3: Funktionsanforderungsbogen

C.5.6 Fehlermanagement

Größere Softwareprojekte sind ohne Fehlermanagement nicht mehr denkbar. Die GenuTEP-Gruppe wird das Web-Forum TWiki [TWi04] als Bugtracking-Tool nutzen. Auf einem Bereich im TWiki wird ein Fehler mit folgenden Daten erfasst

- Modul
- Klasse
- Methode
- Version
- Testnummer
- Fehlerberschreibung
- Prioriät
- Status

Für jeden Fehler wird eine Seite angelegt, die die Fehlerverarbeitung dokumentieren soll.

Literaturverzeichnis

[AES03] AESCULAP AG & CO. KG: *Orthopilot*. http://www.orthopilot.de/, Oktober 2003.

[Aug98] AUGSTEN, N.: *Robuste Schätzung der Fundamentalmatrix*. Doktorarbeit, Technische Universität Graz, 1998.

[B+00] BRONSTEIN et al.: *Taschenbuch der Mathematik*. 5 Auflage, 2000.

[Bal96] BALZERT, H.: *Lehrbuch der Software-Technik I*. Spektrum Akademischer Verlag, 1996.

[Bal98] BALZERT, H.: *Lehrbuch der Software-Technik II*. Spektrum Akademischer Verlag, 1998.

[Bor04] BORLAND: *Together Technologies*. http://www.borland.com/together/, August 2004.

[Bra03] BRAINLAB, MÜNCHEN: *Vector Vision*. http://www.brainlab.com/, Oktober 2003.

[Bur00] BURKHARDT, R.: *UML - Unified Modeling Language: Objektorientierte Modellierung für die Praxis*. Addison-Wesley Verlag, 2000.

[CC93] COHEN, L. und I. COHEN: *Finite-Element Methods for Active Contour Models and Balloons for 2D and 3D Images*. IEEE Trans. Pattern Anal. Mach. Intell., 15(11):1131–1147, 1993.

[Ced03] CEDERQUIST, P.: *CVS Manual*. http://www.cvshome.org/docs/manual/, November 2003.

[CH94] CHRISTY, S. und R. HORAUD: *Euclidean Shape and Motion from Multiple Perspective Views by Affine Iterations*. Technischer Bericht 2421, Inria, Dezember 1994.

[CK03] CROY, A. und I. KABADSHOW: *Computertomographie - CT*. Johannes-Keppler-Gymnasium, Chemnitz, November 2003.

[CT01a] COOTES, T. und C. TAYLOR: *Statistical Models of appearance for Computer Vision*. Technischer Bericht, Universität Manchester, 2001.

[CT01b] COOTES, T. und C. TAYLOR: *Statistical models of appearance for medical image analysis and computer vision*, 2001. In Proc. SPIE Medical Imaging.

[Dav04] DAVIES, R.: *Newmat short introduction*. http://www.robertnz.net/nm_intro.htm/, August 2004.

[Fic01] FICHTINGER, G.: *Fluoroscopy*. GaborEngineering Research Center (ERC) for Computer Integrated Surgical Systems and Technology, 2001.

[FR98] FANG, S. und SRINIVASAN R.: *Volumetric-CSG - A model-based volume visualization approach*. Sixth International Conference in Central Europe on Computer Graphics and Visualization, Seiten 88–95, 1998.

[Fre04a] FREE SOFTWARE FOUNDATION: *GCC Home Page*. http://www.gnu.org/software/-gcc/gcc.html, Januar 2004.

[Fre04b] FREE SOFTWARE FOUNDATION: *GNU Automake*. `http://www.gnu.org/software/automake/automake.html`, Januar 2004.

[G+96] GORTLER, S. et al.: *The Lumigraph*. Computer Graphics, 30:43–54, 1996.

[G+03] GRÜTZNER et al.: *Klinische Studie zur registrierungsfreien 3D-Navigation mit dem mobilen C-Bogen SIREMOBIL Iso-C3D*. http://www.med.siemens.com/medroot/en/-news/electro/issues/pdf/heft_1_03_d/09_gruetzner.pdf, Januar 2003.

[GW92] GONZALES, R. und R. WOODS: *Digital Image Processing*. Addison-Wesley, 1992.

[H+99] HILAL, I. et al.: *Virtual Animation of the Kinematics of the Human for Industrial, Educational and Research Purposes*. Technischer Bericht 2421, Universität Brüssel, 1999.

[Ima03] IMAGING, QUEENSLAND DIAGNOSTIC: *Fluoroscopy*. http://www.qdixray.com.au-/Fluoroscopy/, November 2003.

[Ins03] INSTITUT FÜR TELEMATIK IN DER MEDIZIN: *DICOM - Eine Einführung*. http://-www.iftm.de/forschung/dicom/einfuehrung.htm, November 2003.

[Jur87] JURIS: *Verordnung über den Schutz vor Schäden durch Röntgenstrahlen Datum*. http://www.juris.de/, Januar 1987.

[Jäh02] JÄHNE, B.: *Digitale Bildverarbeitung*. Springer Verlag, 2002.

[Kac37] KACZMARZ, S.: *Angenäherte Auflösung von Systemen linearer Gleichungen*. Bull. Int. Acad. Pol. Sci. Lett., A, 35:335–357, 1937.

[Kan99] KANG, S.: *KDOC - C++ documentation extraction tool*. http://www.ph.unimelb.-edu.au/ ssk/kde/kdoc/, September 1999.

[KC00] KAUFMAN, A. und M. CHEN: *Volume Graphics*. Springer Verlag, 2000.

[KDe03] KDEVELOP: *The KDevelop-Project*. http://www.kdevelop.org/, November 2003.

[KDE04] KDE: *Valgrind - A GPL'd system for debugging and profiling x86-Linux programs*. http://valgrind.kde.org/index.html, August 2004.

[Kra03] KRANKENHAUS BOBLINGEN: *Aufbau einer Knieprothese*. http://www.krankenhaus-bobingen.de/chirurgie/schwerpunkte/unfall/knie/knieteporiginal/aufbau_einer_knie-prothese.htm, Oktober 2003.

[LCN98] LICHTENBELT, B., R. CRANE und S. NAQVI: *Introduction to volume rendering*. Prentice-Hall, Inc., 1998.

[LL94] LACROUTE, D. und M. LEVOY: *Fast volume rendering using a shear-warp factorization of the viewing transformation.* Seiten 451–458, 1994.

[Luo96] LUONG, Q. AND FAUGERAS, O.: *The Fundamental Matrix: Theory, Algorithms and Stability Analysis*. Int. Journal of Computer Vision, 17:43–75, 1996.

[Med03] MEDIVISION ADVANCED SUPPORT SYSTEMS: *Medivison*. http://www.medivision.-ch/, Oktober 2003.

[Mic04] MICROSOFT: *Visio*. http://office.microsoft.com/visio/, August 2004.

[MT96] MCINERNEY, T. und D. TERZOPOULOS: *Deformable models in medical image analysis: a survey*. Medical Image Analysis, 1(2):91–108, 1996.

[MW03] MEDICINE-WORLDWIDE: *Knieendoprothese*. http://www.medicine-worldwide.de/-krankheiten/orthop_erkrankungen/knieendoprothese.html, Oktober 2003.

[Ope04] OPENGL: *OpenGL*. http://www.opengl.org/, Januar 2004.

[Oua95] OUALLINE, S.: *Practical C++ Programming*. O'Reilly & Associates, Inc., 1995.

[PD84] PORTER, T. und T. DUFF: *Compositing digital images*. Seiten 253–259, 1984.

[PG 02] PG 412: *Endbericht der Projektgruppe 412 3D-IVUS View*. Technischer Bericht, Universität Dortmund, 2002.

[PG 04] PG 434: *Zwischenbericht der Projektgruppe 434 GenuTEP*. Technischer Bericht, Universität Dortmund, 2004.

[PMM03] POOLEY, MCKINNNEY und MILLER: *The AAPM/RSNA Physics Tutorial for Residents*. http://radiographics.rsnajnls.org/cgi/content/full/21/2/521, November 2003.

[Pol03] POLIWODA, CH.: *VGL 3.2 User and Reference Manual*. Volume Graphics GmbH, 2003.

[Pot87] POTMESIL, M.: *Generating Octree Models of 3D Objects from Their Silhouettes in a Sequence of Images*. Comp. Vis. Graph. and Image Proc., 40(40):1–29, 1987.

[Qhu04] QHULL: *Qhull*. http://www.qhull.org/, Juli 2004.

[R+94] RUMBAUGH, J. et al.: *Objektorientiertes Modellieren und Entwerfen*. Verlag Carl Hanser u. Prentice-Hall International, 1994.

[Rie97] RIEDEMANN, E. H.: *Testmethoden für sequentielle und nebenläufige Software-Systeme*. B. G. Teubner Verlag, 1997.

[Rup92] RUPRECHT, D. UND MÜLLER, H.: *Image Warping with Scattered Data Interpolation Methods*. Technischer Bericht 443, Universität Dortmund, Fachbereich Informatik, 1992.

[Rup95] RUPRECHT, D.: *Geometrische Deformation als Werkzeug in der graphischen Datenverarbeitung*. Doktorarbeit, Universität Dortmund, 1995.

[Sch03] SCHULZE, MARK A.: *Circular Hough Transform Demonstration*. http://www.markschulze.net/java/hough/, Juli 2003.

[SFE01] STEGMANN, M., R. FISKER und B. ERSBOLL: *Extending and applying active appearance models for automated, high precision segmentation in different image modalities*. Seiten 90–97, Juni 2001.

[SG01] STEGMANN, M. und D. GOMEZ: *A Brief Introduction to Statistical Shape Analysis*, März 2001.

[SHB98] SONKA, M., V. HLAVAC und R. BOYLE: *Image Processing, Analysis and Machine Vision*. PWS Verlag, 1998.

[Sil03] SILICON GRAPHICS INC.: *Standard Template Library Programmer's Guide*. http://-www.sgi.com/tech/stl/, November 2003.

[Smi03] SMITH, S. W.: *Data Compression*. http://www.dspguide.com/datacomp.htm, November 2003.

[SNG01] STEGMANN, M. B., J. C. NILSSON und B. A. GRØNNING: *Automated Segmentation of Cardiac Magnetic Resonance Images*. In: *Proc. International Society of Magnetic Resonance In Medicine - ISMRM 2001, Glasgow, Scotland, UK*, Band 2, Seite 827, April 2001.

[SR01] SUN, Z. und CH. RAYBURN: *Camera Calibration*. Internship with UNR & Ford Motor Company, 2001.

[Ste00] STEGMANN, M.: *Active Appearance Models - Theory, Extensions and Cases*. Doktorarbeit, Technische Universität Dänemark, 2000.

[Ste01] STEGMANN, M.: *Object tracking using statistical models of appearance*. 10th Danish Conference on Pattern Recognition and Image Analysis, 1:136–142, 2001.

[Ste02] STEGMANN, M. B.: *Analysis and Segmentation of Face Images using Point Annotations and Linear Subspace Techniques*. Technischer Bericht, Informatics and Mathematical Modelling, Technische Univerität Dänemark, August 2002.

[Str97] STROUSTRUP, B.: *The C++ Programming Language*. Addison Wesley Longman, 1997.

[Sun03] SUN MICROSYSTEMS: *The Source for Java Technology*. http://java.sun.com/, November 2003.

[sur03] SURFMED: *Computertomographie (CT)*. http://www.surfmed.de/-ComputerTomographie/, November 2003.

[Tiz98] TIZHOOSH, H.: *Fuzzy-Bildverarbeitung*. Springer Verlag, 1998.

[TM96] TRIGGS, B. und R. MOHR: *Projective Geometry for Image Analysis*. Tutorial der ISPRS, Vienna, 1996.

[Tri96] TRIGGS, B.: *Factorization Methods for Projective Structure and Motion*. Proc. Conference on Computer Vision and Pattern Recognition, Seiten 845–851, 1996.

[Tro03] TROLLTECH: *Trolltech - Creators of Qt - The multi-platform C++ GUI/API*. http://-www.trolltech.com/, November 2003.

[TS96] TRIGGS, B. und P. STURM: *A factorization based algorithm for multi-image projective structure and motion*. ECCV, 1065:709–720, April 1996.

[Tur01] TURBELL, H.: *Cone-Beam Reconstruction Using Filtered Backprojection*. Doktorarbeit, Universität Linköping, 2001.

[TWi04] TWIKI: *TWiki - A Web Based Collaboration Platform*. http://www.twiki.org/, August 2004.

[Uni03a] UNIVERSITY OF NOTTINGHAM, SCHOOL OF PSYCHOLOGY: *The DICOM Standard*. http://www.psychology.nottingham.ac.uk/staff/cr1/dicom.html, November 2003.

[Uni03b] UNIVERSITÄT BONN: *Röntgenstrahlung und ihre Gefährlichkeit*. http://www.-med.uni-bonn.de/radiologie/Patienteninformation/Roentgen-Gefahr.html, November 2003.

[Uni03c] UNIVERSITÄT BONN: *Röntgenuntersuchung*. http://www.meb.uni-bonn.de/radiologie/Patienteninformation/% -Roentgen.html, November 2003.

[Uni03d] UNIVERSITÄT HEIDELBERG: *Patienteninformation Knieendoprothese*. http://www.-orthopaedie.uni-hd.de/infopatient/Kuenstl_Gelenke/Knieendoprothese.htm, Oktober 2003.

[Uni03e] UNIVERSITÄT KÖLN: *DICOM 3.0*. http://www.medizin.uni-koeln.de/zde/krz/-projekte/agbildverarbeitung/vortraege/dicom.html, November 2003.

[Uni03f] UNIVERSITÄTSKLINIKUM AACHEN: *Absorption von Röntgenstrahlen*. http://-www.klinikum.rwth-aachen.de/cbt/radiologie/skript/allgemein/Absorbtion.html, November 2003.

[UZ96] UNRUH, P. und H.W. ZELLER: *CE-Kennzeichnung von Medizinprodukten*. VDE-Verlag, 1996.

[Vol03] VOLUME GRAPHICS GMBH: *Volume Graphics - Solutions About Voxels*. http://-www.volumegraphics.com/, Oktober 2003.

[W3C04a] W3C: *Extensible Markup Language (XML)*. http://www.w3c.org/XML/, Juli 2004.

[W3C04b] W3C: *W3C Document Object Model*. http://www.w3c.org/DOM/, April 2004.

[Wah89] WAHL, F.: *Digitale Bildverarbeitung*. Springer Verlag, 1989.

[Web02] WEBER, R. H.: *Einfluss der Scanparameter auf das Abbildungsergebnis bei der CT-Angiographie*. Doktorarbeit, Universität München, 2002.

[XML98] XML.ORG: *Simple API for XML (SAX)*. http://www.xml.org/xml/-resources_focus_sax.shtml, 1998.

[XP97] XU, C. und J. PRINCE: *Gradient Vector Flow: a new external force for snakes*. IEEE Visual Pattern Recognition, Seite 66, 1997.

[YB98] YU, Z. und C. BAJAJ: *Image segmentation using Gradient Vector Diffusion and Region Merging*. IEEE Visual Pattern Recognition, 1998.

[Zha94] ZHANG, Z.: *A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry*. Technischer Bericht 2273, INRIA, 1994.

[Zha96] ZHANG, Z.: *Determining the Epipolar Geometry and its Uncertainty: A Review*. Technischer Bericht 2927, INRIA, 1996.

[Zha98] ZHANG, Z.: *A Flexible New Technique for Camera Calibration*. Technischer Bericht 11, Microsoft Research, 1998.

[ZL01] ZHANG, Z. und C. LOOP: *Estimating the Fundamental Matrix by Transforming Image Points in Projective Space*. Computer Vision and Image Understanding, 82(2):174–180, 2001.